ICBSLP

2nd International Conference on Bangla Speech and Language Processing

27-28 September 2019

Department of Computer Science and Engineering

Shahjalal University of Science and Technology, Sylhet

ICBSLP 2019 is dedicated in loving memory of

Dr. Md. Khairullah

Without whose help ICBSLP 2018 would not have been such a success last year

Proceedings of
2nd International Conference on
Bangla Speech and Language Processing

(ICBSLP 2019)

Department of Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet-3114, Bangladesh

27-28 September, 2019

Contents

Paper Id.	Paper Title
54	Md. Mehadi Hasan, Md. Ariful Islam, ShafkatKibria and Mohammad Shahidur Rahman. Towards Lexicon-free Bangla Automatic Speech Recognition System
59	ShofiUllah, Sagar Hossain and K. M. Azharul Hasan. Opinion Summarization of Bangla Texts using Cosine Simillarity Based Graph Ranking and Relevance Based Approach
60	RajanSaha Raju, PrithwirajBhattacharjee, Arif Ahmad and Mohammad Shahidur Rahman. A Bangla Text-to-Speech System using Deep Neural Networks
62	Md. Akhter-Uz-Zaman Ashik, ShahriarShovon and Summit Haque. Data Set For Sentiment Analysis On Bengali News Comments And Its Baseline Evaluation
63	Md. Hasan, Md. Motaher Hossain, Adnan Ahmed and Mohammad Shahidur Rahman. Topic Modelling: A Comparison of The Performance of Latent Dirichlet Allocation and LDA2vec Model on Bangla Newspaper
64	Md. Ferdous Wahid, Md. Jahid Hasan and Md. ShahinAlom. Cricket Sentiment Analysis from Bangla Text Using Recurrent Neural Network with Long Short Term Memory Model
66	Arid Hasan, FirojAlam, ShammurAbsar Chowdhury and Naira Khan. Neural vs Statistical Machine Translation: Revisiting the Bangla-English Language Pair
69	NafizSadman, AkibSadmanee, M. I. Tanveer, M.A. Amin and A.A. Ali. Intrinsic Evaluation of Bangla Word Embeddings
70	Mehedi Hasan Palash, ParthaProtim Das and Summit Haque. Sentimental Style Transfer in Text with MultigenerativeVariational Auto-Encoder
71	SouravSarker, SyedaTamannaAlamMonisha and MdMahadi Hasan Nahid. Bengali Question Answering System for Factoid Questions: a statistical approach
77	Manjira Sinha, TirthankarDasgupta and Rakesh Dutta. Does Word2Vec encode human perception of similarity ? : A study in Bangla
79	Monisha Biswas and Mohammed MoshiulHoque. Development of a Bangla Sense Annotated Corpus for Word Sense Disambiguation
80	Arnab Sen Sharma, Maruf Ahmed Mridul and MdSaiful Islam. Automatic Detection of Satire in Bangla Documents: A CNN Approach Based on Hybrid Feature Extraction Model

A model of diphone duration for speech synthesis in Bangla

Kamrunnahar Swarna
Department of Computer Science and Engineering,
Shahjalal University of Science and Technology
Sylhet-3114, Bangladesh
swarna.sustcse@gmail.com

M. Shahidur Rahman
Department of Computer Science and Engineering,
Shahjalal University of Science and Technology
Sylhet-3114, Bangladesh
rahmanms@sust.edu

Abstract*—The aim of this paper is to provide an improved duration model for diphones and a diphone selection technique in Bangla text to speech system Subachan for better speech quality. Segment duration is one of the most important factors for the intonation of the generated speech in concatenative synthesis. For this reason, a good duration model is indispensable. To achieve this, we observed diphone durations in a recorded corpus of words, determined the diphone durations for different positions in the words, observed the signals of consonants in the corpus and categorized them accordingly. The results show that the proper durations produce notably better intonation in the generated speech which improves the naturalness and intelligibility of Subachan.

Keywords— ***diphone durations; quality of output speech; speech synthesis; Bangla text to speech***

I. INTRODUCTION

Researchers have found out that diphone duration has a major role in the intonation of the synthesized speech. It is said the pitch and durations are enough to express intonation and emotions in the synthesised speech [1][2]. Intonation plays a major role in the naturalness and intelligibility of a speech synthesis system's output. For these reasons and considering the lack of naturalness in the output of Bangla text to speech system Subachan, a proper duration modelling was always necessary. This motivated our approach. There have been several experiments where researchers found out that the duration of diphones vary in different contexts even though they are stored in a fixed database [3]. The different contexts are for example, the lengthening of diphones in a word's final position and different lengths for the same vowel for different consonants. To illustrate, vowel আ becomes longer when it is with খ than ক i. e. আ is longer in আখ than $_{আক}$. These diphone duration properties were observed in 200 individual words' signals and in the recorded corpus for preparing diphones. Therefore, we created new duration model according to these properties. We also found out that diphones with certain consonants have the same durations when they are paired with the same vowel. Thus, we placed them in the same category. For their duration variations, we created three sets of the same diphones for the three different positions of a word, i.e. first, middle and the end. We implemented the program to get diphones according to their positions. With the new duration model and new set of rules to select diphones, the output speech turned out to be very satisfactory.

II. RELATED WORKS

There exists a previous work on diphone duration modelling [4]. The diphones are lengthier and so they create less natural output speech. There were no mentions of diphone duration variance in different contexts.

III. METHODOLOGY

A. Measurement of Diphone Durations

1) Corpus: There are 4 corpora recorded for diphone preparation.
Among them corpus 4 was chosen because of its intelligibility [4]. But it is slower than necessary. If we remove a few repeating periodic frames from the diphones, they have the same sound but shorter duration. That way, we can achieve both intelligibility and naturalness using this corpus. The long form from the previous model and short form from the new model of the same diphone is shown in Fig. 1.

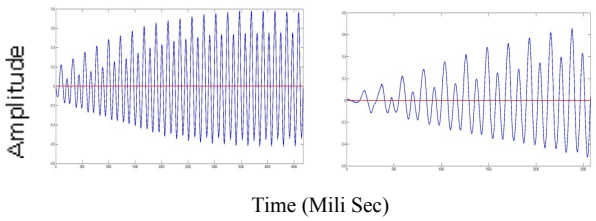


Fig. 1 Repeating vowel frames for diphone . These frames are actually what make the diphones longer, if you reduce some, they get shorter, but the sound quality stays the same. The figure in the left have durations 106ms while the figure in the left has duration of 58ms. Diphone in the right produces much better intonation than the left one.

2) Observing the recorded corpus and determining duration:
We took 200 words which were previously recorded and observed the diphones in them. For example, there is a word কাল with diphones {ক, কআ, আল, ল}. We took the duration of the diphone কআ from this word and from several other words and averaged the durations. The observation led to the revelation- the durations of diphones varied according to their positions in the word. As expected, we found three positions where the duration of diphones varied the most – first, middle and last. The diphones in the middle were relatively smaller in durations than the

978-1-7281-5242-4/19 ©2019 IEEE

diphones in the beginning and end. For example, diphone অর at first position is 95ms, at middle is 61ms and at last is 106ms. We can see that there are significant differences in the same diphone's durations in the middle and in the diphone boundaries. To determine the duration of a diphone, we observed five words containing this diphone in the same position. The results showed that the durations are close, so we averaged the durations to calculate the final duration.

There were some cases where no recorded words were available containing a diphone. In those cases, we synthesized some words which include the diphone and determined the best duration from them.

Moreover, we saw some consonants have the same effect on its adjacent vowel in a diphone. That means their durations are the same if they are with the same vowel in a diphone. For example, কআ and টআ have same durations. It was apparent when their durations were taken for the same position of the word. This makes sense because the signals of those consonants are quite similar in structure and duration. The reason behind this is unaspirated consonants are shorter in length than aspirated consonants. Unaspirated unvoiced consonants (ক, ট, ত, প) have the same diphone durations. Similarly, unaspirated voiced consonants (গ, ড, দ, ব), aspirated unvoiced consonants (খ, ঠ, থ, ফ) and aspirated voiced consonants (ঘ, ঢ, ধ, ভ) have the same diphone durations. The palatal consonant in each category (চ, ছ, জ, ঝ) has different signal structure hence different duration than the rest of the consonants in those categories. Therefore, they were included in different fitting categories.

চ, জ has similarity with য, so they were placed in one category. The nasal consonants (ন, ম) have similarities with ল and fricative unvoiced শ has similar signal to aspirated voiced ঝ। The same was true for many other consonants and we decided to make their separate category with similar signals and durations.

So the consonants were categorized as ক, ট, ত, প/ খ, ঠ, থ, ফ/ গ, ড, দ, ব/ ঘ, ঢ, ধ, ভ/ চ, জ, য/ ছ, স/ ঝ, শ/ ন, ম, ল/ র, ড়/ হ।

Fig. 2 and Fig. 3 represents the signals of consonants of the same category such as ক, ট, ত, প and খ, ঠ, থ, ফ। Observing the signals, we see that they are similar. Durations of the diphones are given in table I, II and III.

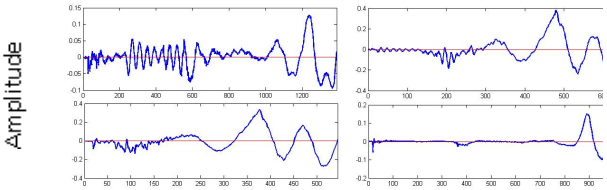


Fig. 2 The signals of the consonants ক, ট, ত, প are clockwise and they are with vowel অ. We can easily see that they are very similar to each other.

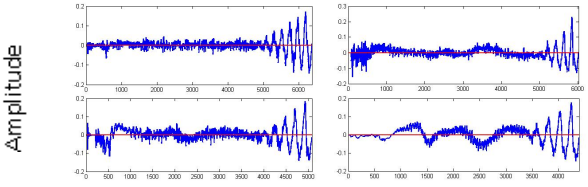


Fig. 3 The signals of the consonants খ, ঠ, থ, ফ are clockwise and they are with vowel অ We can easily see that they are very similar to each other.

B. Implementation of Programs

1) Implementation of diphone selection: We created three sets of diphones instead of one set that was done in the previous model [5]. Previously, the software used to dissect a word into diphones and then pick up the diphones from the same file location address.

We made one file location for each kind of diphone position and placed the diphones with their proper durations in their correct locations. After that, we implemented the program to determine the diphone's position and then fetched diphones of that position from the respective file location.

For example, if কলম is entered as text, the generated diphones and their assigned locations are given in table IV. Then the program fetches diphones from their corresponding locations (First, Middle, End) and plays them. For comparison, old and new systems are shown in skeleton flowchart in Fig. 4 and Fig. 5.

TABLE I DIPHONE DURATION FOR TYPES ({C,CV,C}). THE NUMBERS IN A CELL MEAN FIRST, MIDDLE AND END RESPECTIVELY

	Start	অ	আ	ই	উ	এ	ও	End
[illegible]	25	60,50,82	70,55,66	50,48,48	87,77,84	76,66,108	84,68,102	99
খ,ঠ,থ,ফ	46	110,98,99	116,100,128	138,101,158	75,110,	75,132,148	183,135,141	45
গ,ড,দ,ব	40	80,81,125	100,74,147	80,83,95	98,87,125	59,156,110	93,76,74	67
ঘ,ঢ,ধ,ভ	71	158,157,161	118,95,132	156,127,144	147,80,136	86,117,98	110,96,134	76
চ	46	82,95,136	110,104,104	65,81,98	99,72,90	78,131,104	86,95,101	61
ছ,স	60	193,150,198	141,132,153	114,98,143	107,87,110	170,156,184	154,144,143	83
জ,য	84	145,128,140	120,108,137	102,81,156	99,70,105	107,117,137	94,75,118	61
ঝ,শ	83	161,134,180	148,123,160	95,102,164	121,113,146	88,152,182	156,109,126	112
ন,ম,ল	51	83,88,107	80,56,101	60,67,89	53,67,73	77,56,107	61,91,116	83
র,ড়	40	102,90,115	103,72,80	131,90,48	106,59,105	96,105,98	70,85,78	70
হ	59	75,80,110	111,98,120	55,90,110	110,97,142	114,89,95	111,105,144	132

TABLE II DIPHONE DURATION FOR TYPES ({V,VC,V}). THE NUMBERS IN A CELL MEAN FIRST, MIDDLE AND END RESPECTIVELY.

	Start	ক,ট,ত,প	খ,ঠ,থ,ফ	গ,ড,দ,ব	ঘ,ঢ,ধ,ভ	চ	ছ, স	জ,য	End
অ	71	115,108,144	170,164,191	80,85,91	143,101,153	182,159,171	154,135,156	137,86,161	140
আ	80	130,150,197	184,170,182	89,70,80	111,110,167	140,103,159	151,121,167	142,117,163	172
ই	52	125,118,158	160,151,183	85,79,76	107,105,158	128,92,147	150,135,162	134,111,167	135
উ	55	103,151,160	137,120,141	92,72,103	129,155,160	126,101,156	151,143,177	130,118,173	145
এ	80	184,147,178	139,141,162	78,92,80	134,121,150	142,130,153	123,121,132	115,90,149	137
ও	54	163,150,190	143,154,173	83,85,85	113,152,159	125,134,149	150,121,140	138,133,169	119

2) Adding missing diphones: *In the previous versions of* Subachan, words ending with a vowel would have their closing diphone missing. For instance, আলো would be broken as: {আ, আল, লও} instead of {আ, আল, লও, ও}।

The word would end abruptly for missing ও}. This used to create unnatural output speech. We implemented the program so that the vowel diphone in the end gets called and then prepared those closing vowel diphones which were missing from the previous corpus.

IV. COMPARISON WITH THE PREVIOUS MODEL

In the previous model, the consonants like ক, খ, গ, ঘ were placed in one category, according to the place of pronunciation phonetic rule. In reality, their durations are significantly different from each other. This affected the naturalness of Subachan such that wrong phones were being stressed. On the other hand, we categorized the consonants after observing the durations and their signal similarity. The observation revealed a different phonetic property would be more fitting. We categorized by the classification of consonants according to pronunciation rule. Also we made three sets of diphone duration which contributed to more naturalness of a word.

In our model, some VC (consisting of a vowel and then a consonant) diphones with consonants like ক, ট, ত, প are significantly longer in duration. In the old model, they excluded the tail like part at the end of those diphones. On the surface, it doesn't seem to have any differences, but when those diphones are in the middle of a word, the word turns less intelligible for the missing information from the diphone. We corrected this in the new diphone set which impacts intelligibility and naturalness in a significant amount. Apart from these drawbacks, the old model had lengthier diphones which made the output speech unnatural. Fig. 6 shows the difference of the diphone আক in the previous and new models. In the new model, the burst is included while in the old model it is excluded.

TABLE III DURATIONS FOR THE REMAINING VC DIPHONES

	ঝ,শ	ন,ম,ল	র,ড	হ
অ	147,99,149	83,82,126	92,61,106	130,100,139
আ	139,129,150	84,100,104	77,50,105	149,132,150
ই	92,90,118	90,89,76	79,58,83	107,92,109
উ	160,139,171	98,72,86	61,52,79	133,98,103
এ	100,124,162	97,77,88	68,110,80	102,83,110
ও	161,100,160	111,64,98	69,68,75	117,99,120

TABLE IV HOW DIPHONES ARE GENERATED WHEN কলম IS ENTERED AS INPUT TEXT

First	Middle	End
{ক	অল	অম
কঅ	লঅ	ম}

V. RESULTS AND DISCUSSION

The output from the new duration model produces more intelligible and more natural speech. Also, adding the vowel diphones in the boundaries such as {V, V} increased naturalness notably.

We used a text from the novel "Aranyak" by Bibhutibhushan Bandyopadhyay to create output using the new diphone durations and the old diphone durations. Both outputs were given to 50 listeners and they reported that the new model generated better quality speech. 90% reported that the new output speech has more naturalness than the old one while 10% said that there were no changes. In terms of intelligibility, 95% said that the new output speech were more intelligible than the old one while 4% said there were no changes and the remaining 1% said that the intelligibility was worse. The results are shown in Fig. 7.

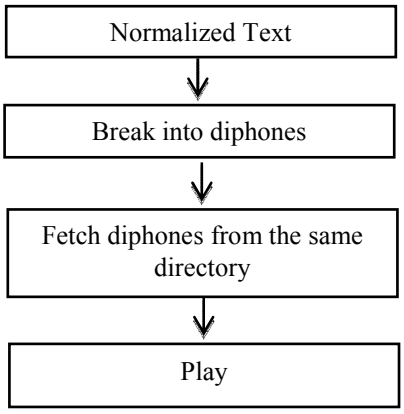


Fig. 4 Old diphone generation and playing technique

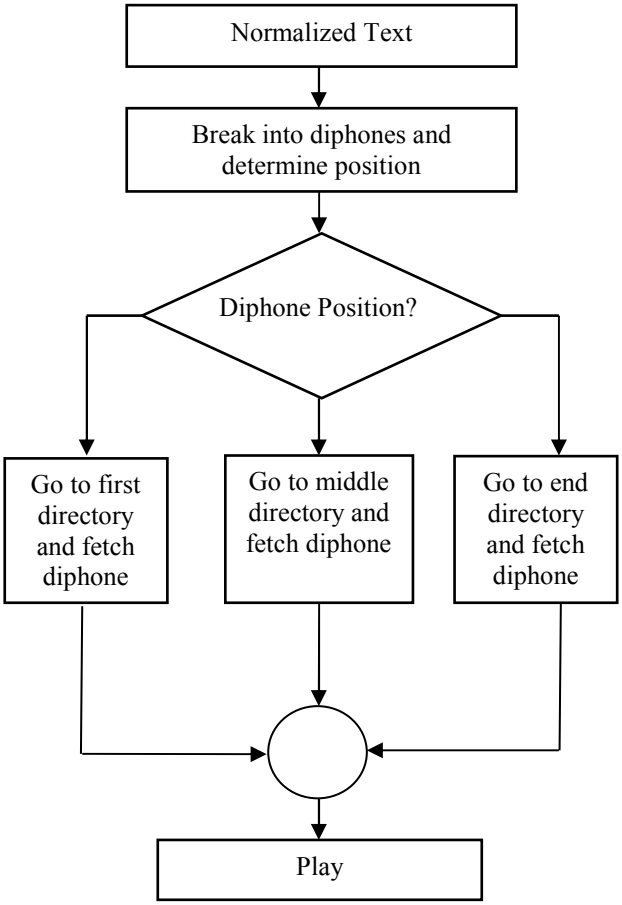


Fig. 5 Improvised diphone generation and playing technique for the new duration model

VI. CONCLUSION

We found a better duration model for Bangla diphones and tested them in Bangla text to speech system Subachan. The output speech has much better quality than the previous duration model. The durations are specified according to the position of diphones in a word and then we implemented the program so that Subachan determines the diphone position and then derives the proper diphone. We also implemented the program so that the vowel diphones at the end are added. They were missing in the previous versions of Subachan. After all those modifications, the generated output speech of Subachan is remarkably improved in terms of naturalness and intelligibility.

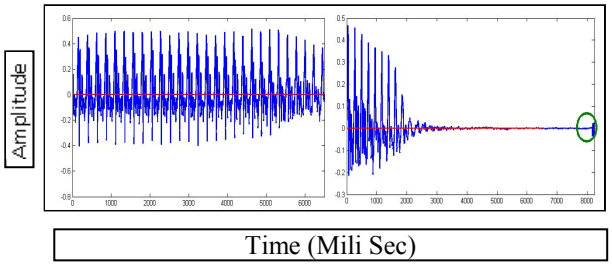


Fig. 6 Left figure shows the old diphone আক which has the end burst missing which is shown in the circled area of the right figure. This is needed to sound properly in the output speech.

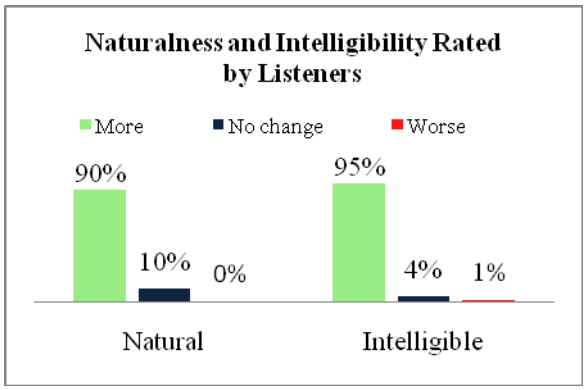


Fig. 7 The performance of new durations reported by the listeners.90% reported that the new output speech has more naturalness than the old one while 10% said that there were no changes. On the other hand, 95% said that the new output speech were more intelligible than the old one while 4% said there were no changes and the remaining 1% said that the intelligibility was worse.

REFERENCES

[1] L. Jiang, "The importance of duration and f0 for generating more natural intonation in the synthetic speech.", 2007.
[2] J. Vroomen, René Collier, and Sylvie JL Mozziconacci. "Duration and intonation in emotional speech.", Eurospeech, 1993.
[3] K. Bartkova and Christel Sorin. "A model of segmental duration for speech synthesis in French." , Speech communication 6.3 ,1987, pp. 245-260.
[4] L. Jahan, Umme Kulsum, and Abu Naser, "Bengali Diphone Duration Modeling for Bengali Text to Speech Synthesis System.", American Academic & Scholarly Research Journal 7.3 ,2015.
[5] M. Masud Rashid, Md Akter Hussain, and M. Shahidur Rahman. "Text normalization and diphone preparation for bangla speech synthesis." Journal of Multimedia 5.6 ,2010, pp. 551-559.

Sentiment Analysis on Movie Review Data Using Machine Learning Approach

Atiqur Rahman, Md. Sharif Hossen

Dept. of Information and Communication Technology
Comilla University, Comilla, Bangladesh
atikriyadict@gmail.com, sharif5613@gmail.com

***Abstract*—At present Sentiment analysis is the most discussed topic which is purposed to assist one to get important information from a large dataset. It centers on the investigation and comprehension of the feelings from the text patterns. It automatically characterizes the expression of feelings, e.g., negative, positive or neutral about the existence of anything. Various sources like medical, social media, newspaper, and movie review can be used in data analysis. Here, we have collected movie review data as well as used five kinds of machine learning classifiers to analyze these data. Hence, the considered classifiers are Bernoulli Naïve Bayes (BNB), Decision Tree (DE), Support Vector Machine (SVM), Maximum Entropy (ME), as well as Multinomial Naïve Bayes (MNB). Our analysis outlines that MNB achieves better accuracy, precision and F-score while SVM shows higher recall compared to others. Besides it also show that BNB Classifier achieves better accuracy than previous experiment over this classifier.**

Keywords— precision, sentiment, classifiers, recall, opinion

I. INTRODUCTION

Internet makes people easier to connect each other. They express their opinion using internet through social media, blog post, movie review, product review site etc. Everyday huge amounts of data are generated by user. Movies are probably the best form of entertainment for mankind and it is common that people watch the movies and express their opinions either in social networking sites. By analyzing movie review data we can learn about the strong and weak point of a movie and tell us if the movie meets the expectation of the user. When a person wants to watch a movie, he first checks the review and rating of the movie. Sentiment analysis (SA) helps in obtaining the review of that movie.

SA is the process of getting valuable fact from a big set of data. It automatically classifies the people opinion as a positive or negative view. According to [1], SA techniques are Sentence, Document, Aspect and User based. The first technique detects the sentiment of each sentence as positive or negative. The second technique detects the sentiment of full document as a single unit. The third technique focuses on all the properties of an entity. Last technique conducts the social interrelation having graph theory for different parities. Machine learning (ML) and Lexicon based techniques are the most common in SA where the first uses training and testing set to categorize data. While the second approaches are used as a dictionary which contain predefined positive and negative words [2]. Here, we have used ML technique to classify data. Using document based SA we find the accuracy of different classifiers where Multinomial Naïve Bayes (NB) has better accuracy (88.50%) than others while Bernoulli NB achieves 87.50%. The organization of the paper is following: The related investigation has been discussed in section II. Section III includes the analysis procedure of SA. Section IV discusses about various types of machine learning algorithm. Section V describes the experimental results. Section VI includes the summary and future endeavor.

II. RELATED WORK

This paper evaluates the opinion on movie review data. SA is handled by natural language processing (NLP) using several levels. Various approaches are available for doing this.

In [2] authors proposed a method using word embedding to create sentiment lexicon by word vector representation. Authors of [3] discussed about the usage of SVM in SA with different source of data. In [4], authors proposed a method based on a sentiment score vector for SVM. In [5] Hu and Liu research was churning out the product features and gave product based summary. Authors of [6] worked on feedback data from global support services survey.

Sentiment analysis has many usages on movie review dataset using different techniques. Authors of [7] broadly categorized SA techniques into ML and lexical-based. Yonas Woldemariam has done SA based on ML and lexicon based approaches. He used apache hadoop framework with lexicon based model [8]. Authors of [9] discussed about entity-level sentiment analysis of issue comment. Authors of [10] discussed about the sentiment lexicon dictionary enrichment based on word2vec. They enlarge the opinion words by using SentiwordNet. In [11], authors deal the view-level SA on e-commerce data. According to [12], SA can be applied to detect the polarity of customer reviews in several dimensions.

According to [13], SA can be done based on combined techniques. ML needs training and testing data set. There are two types of ML methods, namely, supervised method (SM) and unsupervised method (UM). SM is a method in which we at first teach the machine providing some data. It [14] has several algorithms to conduct the classification technique based on the trained data which are. Naïve Bayes (NB), SVM, ME, DE etc. There is no training data in unsupervised learning where data are unlabeled. Author of [15, 16] design advanced NB algorithm with parts-of-speech tagging. He also claim that NB algorithm is costly wastes a lot of resources. Authors of [13] discussed the NB and complement NB classifier algorithm on hadoop framework. According to [17] SVM classifier was also conduct to find the public user counsel on products. NB and SVM were used [18] on various medical forum data. Authors of [19] have done the sentiment analysis on Bengali data about Bangladesh cricket team using support vector machine. In [20], they done the sentiment analysis using KNN, Bagging, COCR, NB and

978-1-7281-5242-4/19 ©2019 IEEE

Decision tree classifier. Authors of [21] used deep learning recurrent neural network method and decision tree for movie review data.

Two kinds of lexicons are available [22]. The 1st one is corpus based lexicon and second one is dictionary based lexicon. Corpus based lexicon, like as SenticNet [23] can acquire more appropriate sentiment outcome as it is context oriented instead of accord of words oriented. Semantic oriented concept was applied based on a concept net lexicon. In [24] the authors show a statistical approach to find the sentiments. Authors of [25] show the 2 dictionaries. The 1st one is word dictionary and the second one is topic modeling. According to [14], a small set of counsel words are culled manually with acquainted orientations in dictionary based approach. This paper follows the machine learning technique for sentiment analysis.

III. Analysis Procedure

In this section, we discuss the analysis procedure. Figure 1 show the steps and techniques used in this paper for classification our text.

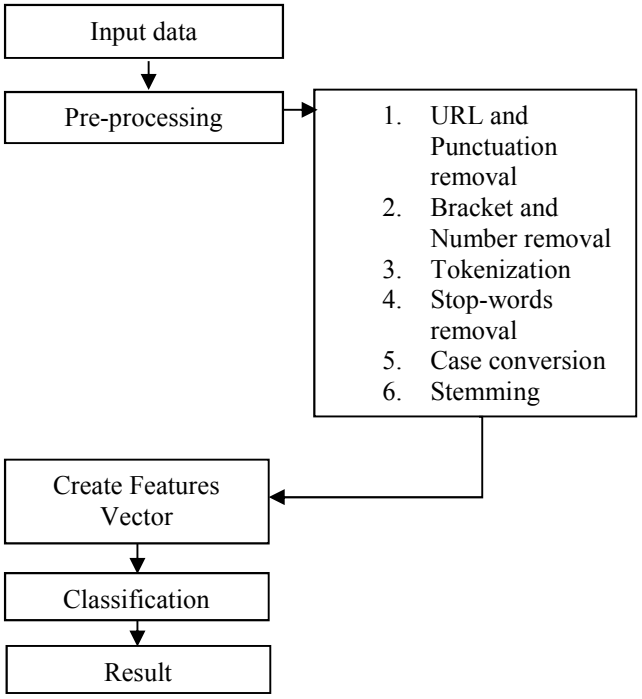


Fig. 1: Steps for classification in SA

First of all we created our dataset from the movie review data (Data source are mentioned in reference section [26]). We collected two thousands movie review posts where there exists equal number of positive and negative reviews. Then, we followed those steps for our sentiment classification.

A. Text Preprocessing

a) URL Removal: URL is a sharing link which is also known as HTML tag. Many texts contain URL or HTML tag. First of all we remove URL from the text.

b) Bracket and Number Removal: Bracket and numbers have no meaning in sentiment classification. These are treated as noise. We need to clean bracket and number from the text.

c) Tokenization: We divide our textual data into smaller components. This can be done by using tokenization. Tokenization provides us to turn text into sentences and sentences into words.

d) Omitting Punctuation: Quotation mark, semi-colon, colon, etc. are omitted as there are no data.

e) Case conversion: To remove the distinction between "Review" and "review" it is done.

f) Omitting stop words such as "I", "it", "you", "a", "an", "the" since they have no meaning in sentiment classification. We should remove those words from the input text.

g) Stemming: It is the method to reform the inflected words and removes derivational affixes from a word.

B. Feature Vector Creation

Feature is a measurable characteristic of a phenomenon. We classify each review as positive or negative. Every review is considered as a simple document. In unigram model, every document is denoted by its primary words where positive and negative primary words are used for respective positive and negative document. We also added parts of speech (PoS) tags to get the more accurate sentiment. Emotions are determined by opinion words. Positive and negative sentiments are maintained by respective sentiment scores. For this reason, these words are instrumental feature to sentiment analysis. How many features are there is termed as dimensionality. Positive and negative keywords are signed as pos and neg respectively for every document. Then, PoS are included with sentence. For this reason every word has a feature. Depending on the more variety of dimensions classifiers shows worst outcome.

IV. Considered Classifiers

A. NB Classifier

NB [27] is a SM which is the easiest and most recognizable used classifier. It considers that every feature is distinct from one another. NB classifiers are a collection of classifications algorithm which is not a standalone algorithm but a family of algorithm. The mathematical expression is as follows:

$$P(x|Y) = \frac{P(Y|x)P(x)}{P(Y)}$$

Where

$$P(Y|x) = P(y_1|x)P(y_2|x) \dots P(y_n|x)$$

Here, X is class variable and Y is a dependent feature vector. P(x) and P(x|Y) denote the respective priori and posteriori probability of x and Y.

We use two Bernoulli and Multinomial NB techniques where Multinomial NB is good for when features describe discrete frequency counts (e.g. word counts). If we want to estimate P(W | Z), the decision rule for this classifier become

$$P(W|\ Z) = \frac{count(this\ W\ in\ class) + 1}{count(all\ W\ in\ class) + count(all\ W)}$$

where W denotes sentiment. BNB is good for making predictions from binary features. Mathematically,

$$P(a_j \mid b) = P(j \mid b)a_j + (1 - P(j \mid b))(1 - a_j)$$

which differs from MNB approach.

B. SVM

SVM is another SM technique which is used to figure out each raw fact as a dot for fixed dimensions of features. Then, we choose a hyper-plane between two classes. There are several hyper-planes, but we choose one that maximizes the margin between the two classes. According to [14], text categorization is consummately appropriate for SVM due to the meager idea of content, where not many highlights are unimportant, yet they will in general be associated with each other and by and large planned into straightly particular classes.

C. ME Classifier

ME is an algorithm which uses probabilistic concept and does not guess that the characteristics are conditionally autonomous of each other. It is utilized when we cannot accept the restrictive autonomy of the features. Using the training data it makes a model which can prefer the trained data having higher entropy after a much time than other classifiers.

D. DT Classifier

DT differentiates those records having many features checking the property from the root and vertex in a tree. All terminal vertexes are assigned a class label pos or neg. The checker on property is the occurrence or non-occurrence of at least one word. Tree is partitioned until there exists least number of records.

V. Experimental Result and Analysis

A. Training and Testing Data

After creating feature vector we apply ML techniques for classification. We take 1400 and 600 movie reviews for training and testing data set where first sets are trained by a classifier and the accuracy of that is calculated using the second. We use five types of main classifiers, namely, MNB, BNB, SVM, ME and DT). All classifiers are implemented by using python. The dataset contains 2000 movie review where 1000 is negative and remaining is positive. We use the terms, namely, true positive (U), false positive (V), true negative (X), false negative (Y) for analysis. Here, first and second terms indicate that the review is really positive and negative respectively but both are featured as positive term. Third and fourth terms indicate that the review is really negative and positive respectively but both are featured as negative term. Accuracy, recall, precision, and F-score are determined from the above terms. Table 2 shows the analysis got from each classifier for data with label.

Table 1. Classifiers with four terms for the 2000 review

Method	U	V	X	Y
Multinomial NB	250	19	281	50
Bernoulli NB	259	34	266	41
SVM	268	44	256	32
Maximum Entropy	254	190	110	46
Decision Tree	247	66	234	53

Table 2. Performance statistics of several classifiers

Method	Accuracy	Precision	Recall	F-score
Multinomial NB	88.50%	92.94%	83.33%	87.87%
Bernoulli NB	87.50%	88.40%	86.33%	87.35%
SVM	87.33%	85.90%	89.33%	87.58%
Maximum Entropy	60.67%	57.21%	84.67%	68.28%
Decision Tree	80.17%	78.91%	82.33%	80.58%

Figure 2 shows the graphical representation of accuracy, precision and recall. We also show the recall versus precision graph for better comparison in Fig 3. From Fig. 2 and Table 2, we see that Multinomial NB has better accuracy compared to others. It obtains an accuracy of 88.50% while Bernoulli NB obtains 87.50%, SVM with 87.33%, Maximum Entropy with 60.67% and Decision tree with 80.17%. Multinomial NB also has high precision and F-score (Table 2), but the SVM has higher recall. Besides it also shows that Bernoulli Naïve Bayes Classifier achieves better accuracy than previous experiment over this classifier [27]. It has the good precision, recall and f-score Maximum Entropy classifier shows the low performance than other classifier. It has the low precision and f-score compare to the others. But, the precision is higher than Multinomial NB and Decision Tree. The above result shows the quality of features vector selected for movie review data.

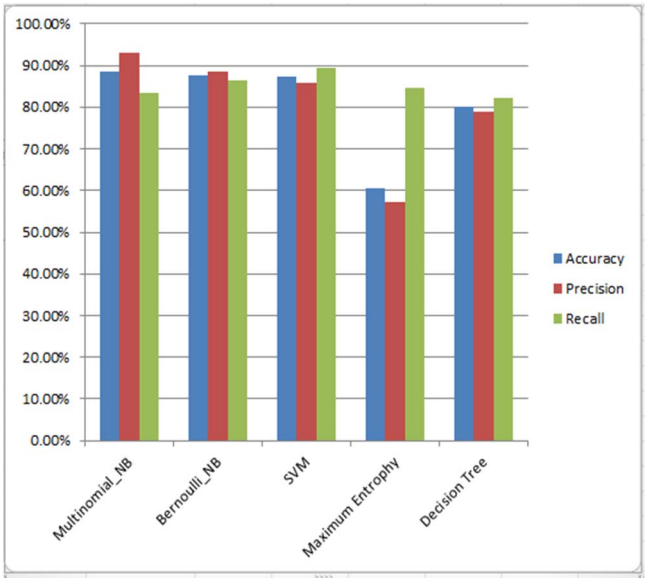


Fig. 2: Performance of difference classifiers

Every classifier is sensitive to parameter optimization. Although the result shows that Multinomial Naïve Bayes classifier is better than SVM, this is only true for selected parameters because multinomial NB show the worse results when training dataset is small.

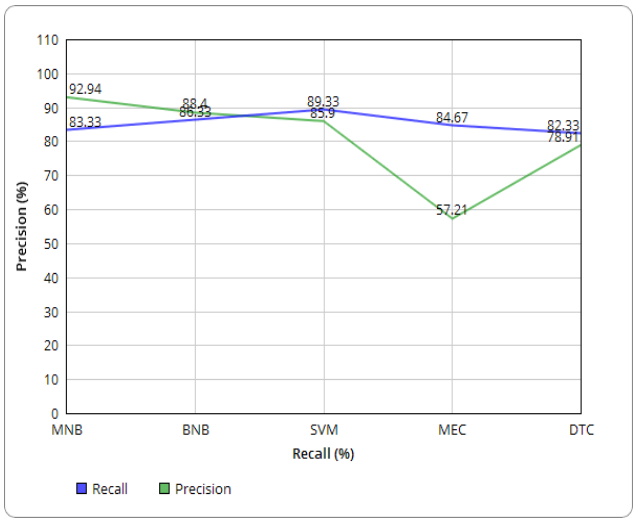


Fig.3: Classifiers for precision vs recall

VI. Conclusion and Future Plan

Sentiment analysis is very essential to understand the expression of feelings about anything like product, social media etc. It can be done by lexicon (LN) and machine learning (ML) approaches. LN can fail to calculate the score of expression if a word not found in the dictionary. While, ML is easier and more efficient but it requires labeled data. In this paper, we use ML approach for polarity classification on movie review data. This approach divides the dataset into two sets, i.e., train and test set. First of all a data set is collected from the movie review site. Next, we perform pre-processing on data by using NLP tool. Then, after creating features vector the data set is trained using ML classifiers, namely, Multinomial NB, Bernoulli NB, SVM, Maximum Entropy and Decision Tree classifiers which are tested using test dataset. Finally, we show our experimental results which present that the accuracy (88.5%) of Multinomial NB is better than others.

In near future, we would like to extend this work using deep learning approaches.

References

[1] C. Tan, L. Lee, J. Tang, L. Jiang, M. Zhou, and P. Li, "User-level sentiment analysis incorporating social network," In Proc. of ICKDDM, IEEE, pp. 1397-1405, 2011.

[2] X. Fan, X. Li, F. Du, Xin Li, Mian Wei, "Apply word vectors for sentiment analysis of APP reviews," In Proc. of ICSI, IEEE, 2016.

[3] T. Mullen, N. Collier, "Sentiment analysis using support vector machines with diverse information sources", In Proc. of ICEMNLP, pp. 412-418, 2004.

[4] S. Naz, A. Sharan, N. Malik, "Sentiment classification on twitter data using support vector machine," In Proc. of ICWI, 2018.

[5] M. Hu and B. Liu, "Mining and summarizing customer remarks," In Pro. of ICKDDM, IEEE, pp. 168–177, 2004.

[6] M. Gammon, "Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis," In Proc. of ICCL, pp. 841-847, 2004.

[7] B. Pang, L. Lee, and S. Vaithyanathan "Thumbs up?: sentiment classification using machine learning techniques," In Proc. of ICEMNLP, pp. 79-86, 2002.

[8] Y. Woldemariam, "Sentiment analysis in a cross-media analysis framework," In Proc. of ICBDA, IEEE, 2016.

[9] J. Ding, H. Sun, X. Wang, X. Liu, "Entity-level sentiment analysis of issue comments," In Proc. of IWEASE, IEEE, 2018.

[10] E. M. Alshari, A. Azman, S. Doraisamy, N. Mustapha, M. Alkeshr, "Effective method for sentiment lexical dictionary enrichment based on word2vec for sentiment analysis," In Proc. of ICIRKM, Malaysia, 2018.

[11] S. Vanaja, M. Belwal, "Aspect-level sentiment analysis on e-commerce data," In Proc. of ICIRCA, 2018.

[12] P. Porntrakoon, C. Moemeng, " Thai sentiment analysis for consumer's review in multiple dimension using sentiment compensation technique.," In Proc. of ICEECTIT, 2018.

[13] B. Seref, E. Bostanci, "Sentiment analysis using naïve bayes and complement naïve bayes classifier algorithms on handoop framework," Int. Symp. on Multisciplinary Studies and Innovative Technologies, 2018.

[14] W. Medhat, A. Hassan, "Sentiment analysis algorithms and applications: Asurvey," Shams Engineering, vol. 5, pp. 1093–1113, 2014.

[15] F. Xianghua, L. Guo, G. Yanyan, and W. Zhiqiang, "Multi-aspect sentiment analysis for chinese online social reviews based on topic modeling and hownet lexicon," Knowledge-Based Sys., vol. 37, pp. 186-195, 2013

[16] Y. Wang, "Advanced naïve bayes algorithm design with part-of-speech tagger on sentiment analysis," In Proc. of Int. Conf. on Computer System, Electronics and control, 2017.

[17] H. Cho, S. Kim, J. Lee, and J.-S. Lee, "Data-driven integration of multiple sentiment dictionaries for lexicon-based sentiment classification of product reviews," Knowledge-Based Sys., vol. 71, pp. 61–71, 2014.

[18] T. Ali, D. Schramm, M. Sokolova, and D. Inkpen, "Can i hear you? sentiment analysis on medical forums," In Proc. of ICNLP, Asian Federation of Natural Language Processing, Nagoya, Japan, pp. 667–673, 2013.

[19] S. A. Mahtab, N. Islam, M. Rahman, "Sentiment analysis on bangladesh cricket with support vector machine," In Proc. of ICBSLP, 2018.

[20] T. P. Sahu, Sanjeev Ahuja, "Sentiment analysis of movie review: A study on feature selection & classification algorithm," In Proc. of ICMCC, 2016.

[21] A. S. Zharmagambetov, A. A. Pak, "Sentiment analysis pf a document using deep learning approch and decision trees," In Proc. of ICECC, 2015.

[22] N. El-Fishawy, A. Hamouda, G. M. Attiya, and M. Atef, "Arabic summarization in twitter social network," Ain Shams Eng., vol. 5, no. 2, pp. 411–420, 2014.

[23] E. Cambria, A. Livingstone, and A. Hussain, "The hourglass of emotions," Cognitive behavioural sys., Springer, pp. 144–157, 2012.

[24] A. Hogenboom, F. Boon, and F. Frasincar, "A statistical approach to star rating classification of sentiment," Management Int. Sys.. Springer, pp. 251–260, 2012.

[25] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexiconbased methods for sentiment analysis," Comp. linguistics, vol. 37, no. 2, pp. 267–307, 2011.

[26] Movie Review 2000 dataset, https://github.com/riyadatik/Sentiment-Analysis-on-Movie-Review-Data/blob/master/Data%20set.xlsx, 2019.

[27] V. Narayanan, I. Arora, A. Bhatia, "Fast and accurate sentiment classification using an enhanced Naïve Bayes model," Int. Data Eng. and Aut. Learning, Lec. Notes. in Com. Sci., vol. 8206, pp 194-201, 2013.

Basic to Compound: A Novel Transfer Learning Approach for Bengali Handwritten Character Recognition

Sakib Reza*, Ohida Binte Amin†, and M.M.A. Hashem‡
Department of Computer Science and Engineering
Khulna University of Engineering & Technology (KUET)
Khulna 9203, Bangladesh
sakibreza1@gmail.com*, ohida.amin2010@gmail.com†, hashem@cse.kuet.ac.bd‡

***Abstract*—Transfer learning is widely used in various character recognition tasks. In this paper, we propose a transfer learning approach with convolutional neural network (CNN) for Bengali handwritten character recognition. When children learn the Bengali scripts, they first learn basic characters (vowels and consonants) and then go for compound characters (consonant conjuncts). Without prior knowledge of basic characters, it would be quite difficult for them to learn compound characters. In our approach, the machine mimics this human child learning process. Our study shows that CNN trained on basic characters is well capable of recognizing compound characters with minimal retraining. It performs better and also trains much faster than CNN fully trained on compound characters. Similarly, CNN trained on digits easily recognizes basic characters with a short period of training. Furthermore, pretrained CNN consistently outperforms the randomly initialized CNN while training only last few layers.**

***Index Terms*—Bengali Scripts; Convolutional Neural Network; Handwritten Character Recognition; Transfer Learning;**

I. Introduction

Bengali is the only official language in Bangladesh and one of the official languages in India. Many official documents in these regions have been written in Bengali over the years and now it becomes really necessary to preserve those documents by digitizing. In this situation, Bengali handwritten character recognition (HCR) comes in action. In the recent past, some great research works are executed in the field of Bengali handwritten character recognition, but still there remains a lot of challenges.

Bengali scripts are mainly divided into basic characters (vowels and consonants), the compound characters (consonant conjuncts), digits and some other symbols. From these types, recognizing the compound characters is found to be the most critical task in Bengali handwritten character recognition. In this study, we take help from an analogy to solve this particular character recognition challenge. When children learn Bengali scripts from their teachers or parents, first, they learn basic characters and then go for the compound characters. As compound characters are constructed with two (or more) basic characters (figure 1), it becomes easy for a child to learn the compound characters with the prior knowledge of the basic characters. Observing the fact, we designed a learning technique for the machine which mimics the learning process of a human child in the task of compound character recognition.

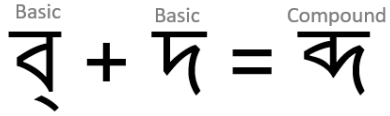


Fig. 1: Example of a Bengali compound character constructed with two basic characters.

In this paper, we present a new transfer learning approach for classifying Bengali handwritten characters. We constructed a convolutional neural network (CNN) architecture and utilized it for our transfer learning analysis. In the first experiment, we trained the CNN with the basic character (source task) and transfer the trained weights for the compound character classification task (target task). This experiment shows us that a pretrained model with minimal retraining can perform better than a randomly initialized model trained for a long time. Again, we conducted another similar transfer learning experiment considering the digits as the source data and the basic characters as the target data. This analysis also validates the efficiency and effectiveness of our designed transfer learning approach.

A. Related Work

In recent years, remarkable progress has been made in the domain of deep learning and its application. Because of this advancement, deep learning based transfer learning approaches have become popular for the character recognition tasks.

D. C. Ciresan et. al. [1] proposed a transfer learning approach for Latin and Chinese character recognition with convolutional neural networks. In this study, they trained a CNN on Latin digits and transferred it for recognizing uppercase Latin letters. They conducted a similar analysis taking Chinese characters as source data and Latin characters as target data; also vice-versa. This study founds that using transfer learning approach a target character classification task can be performed perfectly with minimal retraining.

978-1-7281-5242-4/19/$31.00

In another similar research, A. K. Tushar et. al. [2] introduced a convolutional neural network based transfer learning technique for Bengali, Hindi, and Urdu numeric characters. Here, they considered numerals of one language as the source data and numerals of another language as the target data. This research outcome depicts that model pretrained on one numeric character type can perform well on another type of numeric characters with minimal retraining. For Bengali handwritten character classification, S. Chatterjee et. al [3] designed a method with ResNet50 CNN model [4] pretrained on ImageNet dataset [5]. They evaluated the method on the BanglaLekha dataset (comprising basic, compound and digit characters) [6] and produced a state-of-the-art accuracy with minimum training epochs.

B. Outline

The outline of this paper is as follows. In section II, we describe the proposed method including the data, convolutional neural network architecture and transfer learning approach. section III discusses the experimental results of our designed technique. Finally, section IV concludes the paper.

II. Materials and Method

A. Data

Ekush [7] is one of the most enriched publicly available datasets of Bengali handwritten characters containing over 300,000 instances. Here, every instance is a grayscale image with a resolution of 28×28. The specialty of this dataset is that it contains a few compound character classes of Bengali language along with all Bengali alphabets, digits, and modifiers. For our experimental purpose, we took 30,830 digits of 10 classes, 155,570 alphabets of 50 classes, and 151,607 compound characters of 52 classes from this source dataset. Figure 2 shows the character samples of all the classes used in this study.

B. Convolutional Neural Network

We developed a convolutional neural network (CNN) architecture for conducting the transfer learning experiments. The overall architecture of our developed CNN model is visually described in figure 3. It consists of eight layers - three convolutional layers, three max-pooling layers, and two fully connected (dense) layers.

1) Convolutional Layer: A convolutional layer comprises a filter whose weights require to be tuned through training. The size of the filter is smaller than that of the input map. The filter is convolved with the input map to compute an activation map composed of neurons. Equation 1 shows the convolution operation in a convolutional layer.

$$x_{ij}^{l} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \omega_{ab} \cdot y_{(i+a)(j+b)}^{l-1} \tag{1}$$

Here, x^l is the pre-nonlinearity input of the current layer, ω is a filter of $m \times m$ size and y^{l-1} is the map of the previous layer. To produce the output map y^l, the x^l is passed through a non-linear activation function σ as shown in equation 2.

$$y_{ij}^{l} = \sigma(x_{ij}^{l}) \tag{2}$$

2) Pooling Layer: A pooling layer is generally associated between two consecutive convolutional layers. The pooling layer down-samples the representation to reduce the number of parameters and computation. In our architecture, the max-pooling layer is applied.

3) Fully connected Layer: After several convolutional and max-pooling layers, the final rationalizing in a CNN is performed with fully connected layers. Through fully connected layers, every neuron in one layer is connected to every neuron in another layer. The flattened feature vector obtained from the previous layer goes through a fully connected layer to produce a class label as an output.

4) Proposed Architecture: The proposed CNN architecture starts with a convolutional layer with 32 kernels of size 3×3, it filters a 28×28 image provided as input. Then, the output of the first layer is max-pooled in the second layer. Similar to the previous layers, the "convolutional layer + max-pooling layer" block appears two more times. The last two convolutional layers (third and fifth layers) consist of 64 kernels of size 2×2 and 128 kernels of size 3×3 respectively. Finally, the architecture finishes with two fully connected (dense) layers. The first fully connected layer has 256 neurons and the last fully connected layer has 10/50/52 neurons depending

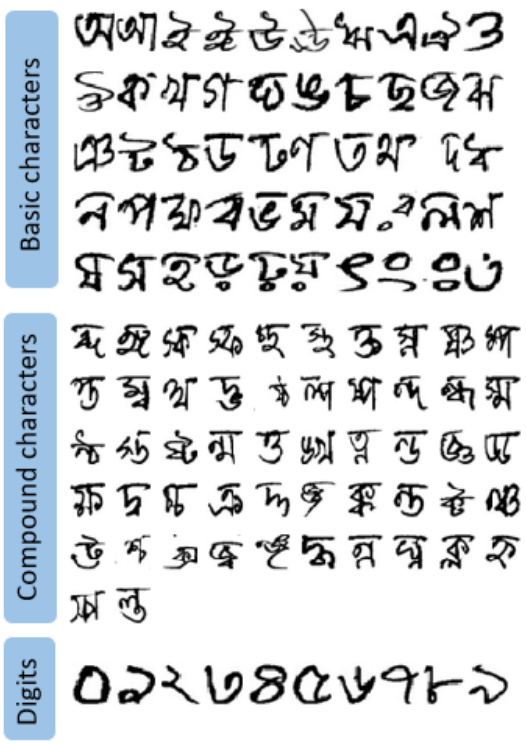


Fig. 2: Samples of all the Bengali character classes used in this study (character images are visualized in binary color-map).

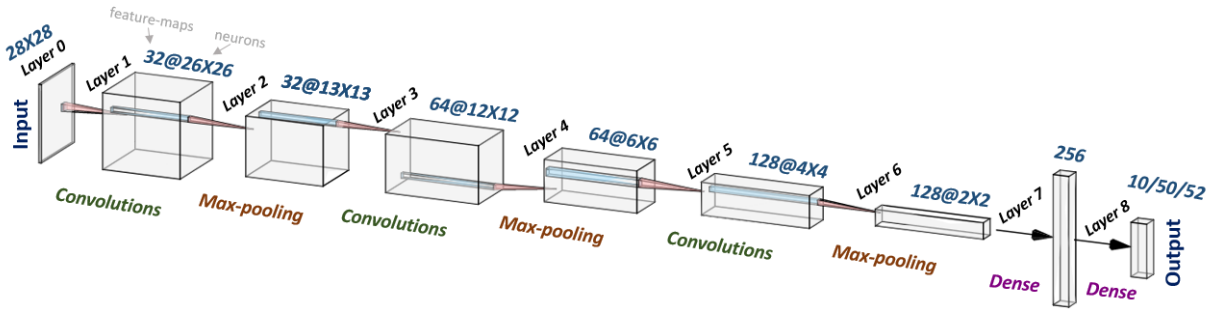


Fig. 3: A visual representation of the proposed convolutional neural network (CNN) architecture.

on the context if it is classifying digits, basic characters or compound characters. Neurons of the last layer with the softmax activation function produce the output class labels. The rectified linear unit (ReLU) non-linear function is applied to the output of every convolutional and fully connected layer except the last one.

C. Transfer Learning

Transfer learning using convolutional neural network (CNN) is a powerful approach to perform an image classification task. In CNN, first few layers extract some generic feature of input data and the features extracted from the last few layers become more specific with the particular task [8]. Therefore, in case of a new classification task, it is quite reliable to reuse the starting layers and train only the finishing layers. In our study, the transfer learning process starts with training our designed CNN on the source task. The training procedure terminates achieving the lowest possible validation error. To use this pretrained CNN with the target data, the output layer is altered with the number of neurons required for the particular classification task (e.g. if we train our net on basic characters and transfer it for compound characters classification, the number of neurons in the output layer will change from 50 to 52). Then, we initialize the weights randomly for the modified output layer; the weights of other layers remain unchanged. Finally, the CNN with pretrained weights is retrained on the target task. Figure 4 depicts the approach for the transfer learning from basic characters to compound characters; the procedure is also the same for the transfer learning from digits to basic characters.

For evaluating our transfer learning approach, we froze the first n layers and trained the rest of the layers. Increasing the value of n gradually from zero, we calculated the error rate of the target task. Here, the possible value of n is 0, 2, 4, 6, 7 as only the convolutional and fully connected layers are trainable; max-pooling layers do not have any weight to train. To check the performance of our approach, we compared the error rates of the pretrained nets with the randomly initialized nets.

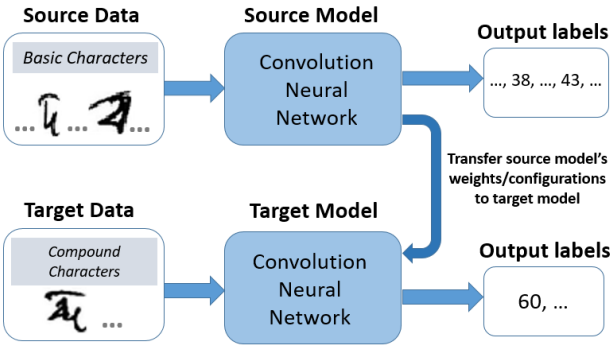


Fig. 4: Transfer learning process using basic characters as source data and compound characters as target data.

III. Experiments

For evaluating the effectiveness of the proposed method, we performed two different experiments. The first one is transfer learning from basic to compound characters and the second one is transfer learning from digits to basic characters. These experiments are performed on a computer configured with Intel Core i5-7300HQ CPU (2.5GHz), 8GB DDR4 RAM and NVIDIA GeForce GTX 1050 GPU. For every individual experiment, the dataset is divided into a training set (70%) and a test set (30%).

A. From basic to compound characters

For the Bengali language, the compound character is the most complicated type of symbols and the basic character (alphabet) is comparatively a simple type. Therefore, we choose the basic characters as the source data and the compound characters as target data in our transfer learning experiment.

This experiment verifies that a pretrained CNN trained on the last few layers can achieve a better outcome than a randomly initialized CNN. The results listed in table I show that the pretrained CNN trained from the 5^{th} layer achieves an error rate of 9.22%, which is lower than the error rate for the randomly initialized net trained from the 1^{st} layer (9.69%). Besides, training only the last two and very last layers of the pretrained model achieves satisfactory error rates of 12.26%

and 15.17% respectively. On the other hand, training the last few layers of the randomly initialized net produces ridiculous results.

In addition, a pretrained CNN utilizing transfer learning technique can converge to a competent state faster than a CNN started its training from scratch. Figure 5 shows that the pretrained CNN achieves a low error rate with a few training epochs. On the contrary, even after training for a long time randomly initialized net fails to achieve a low error as the pretrained CNN.

TABLE I: Test errors for convolutional neural network model pre-trained on basic characters and transferred to compound characters

Initialization	Layer first trained from				
	1	3	5	7	8
Random	9.69%	10.29%	12.05%	27.8 %	53.91%
Basic	8.76%	8.53%	9.22%	12.26 %	15.17 %

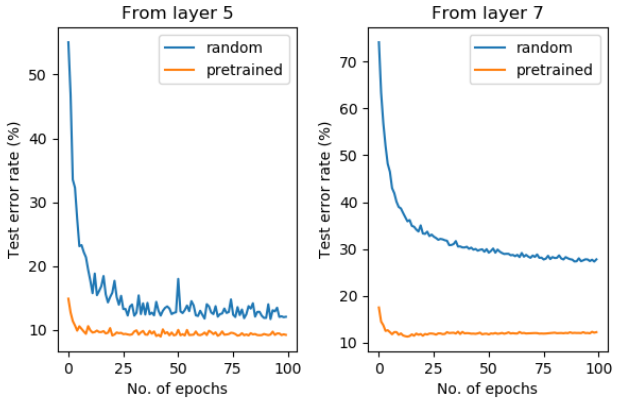


Fig. 5: Test error rates (%) on compound characters as a function of training epochs for randomly initialized CNN (blue) and CNN pretrained on basic characters (orange). the CNN is retrained on compound characters starting from the fifth (left plot) and seventh (right plot) layer, respectively.

B. From digits to basic characters

Similar to the previous experiment, transfer learning from digits to basic characters can be another useful construct. The digit is the simplest type of symbol in Bengali scripts and the basic character is also simple but comparing to digit, a more complex type. For this reason, we choose the digits as the source data and the basic characters as the target data for the transfer learning analysis.

This experiment exhibits quite a similar outcome as the previous one. Table II shows that CNN pretrained with digit data achieves error rates of 10.18% and 16.82% after only training last four and two layers respectively, which is a satisfactory result considering a very short training time. In contrast, CNN initialized with random weights produces unacceptable error rates while training only last few layers. Furthermore, in figure 6, the two plots compare the training progress of the pretrained and the randomly initialized CNN trained from 5^{th} and 7^{th} layers accordingly. This shows us that pretrained CNN reaches to a constant low error rate much quicker than the randomly initialized CNN.

TABLE II: Test errors for convolutional neural network model pre-trained on digits and transferred to basic characters

Initialization	Layer first trained from				
	1	3	5	7	8
Random	6.52%	6.78%	7.57%	18.77 %	41.16%
Digits	7.09%	6.86%	7.41%	10.18 %	16.82 %

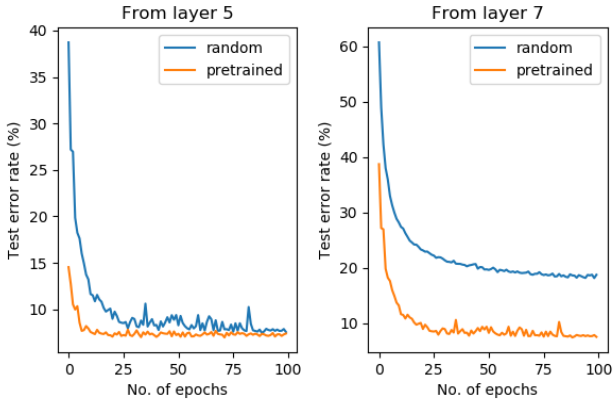


Fig. 6: Test error rates (%) on basic characters as a function of training epochs for randomly initialized CNN (blue) and CNN pretrained on digits (orange). the CNN is retrained on basic characters starting from the fifth (left plot) and seventh (right plot) layer, respectively.

IV. Conclusion

In this paper, the proposed transfer learning method is proven to be effective in the task of Bengali handwritten character recognition. Our study proves that transfer learning from a simple type of characters to a complicated type of characters is certainly acceptable. A CNN model pretrained on basic characters, retraining on the compound character classification task reaches a competent level of error with a short period of training. Similarly, transfer learning from digits to basic characters is also discovered to be a compelling concept. Furthermore, this presented method can also be effective in such a situation where there is not enough data available for a classification task. Utilizing prior knowledge from one type of Bengali scripts, a pretrained model can perform accurately on other types with a small amount of target data and least training. In the future, we may evaluate our proposed method on other available Bengali character datasets for further validation.

REFERENCES

[1] D. C. Cireşan, U. Meier, and J. Schmidhuber, "Transfer learning for latin and chinese characters with deep neural networks," in *The 2012 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2012, pp. 1–6.

[2] A. K. Tushar, A. Ashiquzzaman, A. Afrin, and M. R. Islam, "A novel transfer learning approach upon hindi, arabic, and bangla numerals using convolutional neural networks," in *Computational Vision and Bio Inspired Computing*. Springer, 2018, pp. 972–981.

[3] S. Chatterjee, R. K. Dutta, D. Ganguly, K. Chatterjee, and S. Roy, "Bengali handwritten character classification using transfer learning on deep convolutional neural network," *arXiv preprint arXiv:1902.11133*, 2019.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[5] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[6] M. Biswas, R. Islam, G. K. Shom, M. Shopon, N. Mohammed, S. Momen, and A. Abedin, "Banglalekha-isolated: A multi-purpose comprehensive dataset of handwritten bangla isolated characters," *Data in brief*, vol. 12, pp. 103–107, 2017.

[7] A. S. A. Rabby, S. Haque, S. Islam, S. Abujar, and S. Hossain, "Ekush: A multipurpose and multitype comprehensive database for online off-line bangla handwritten characters," 01 2019.

[8] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3320–3328. [Online]. Available: http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf

Handwritten Bengali Number Detection using Region Proposal Network

Shaharat Tajrean
Institute of Information Technology
Jahangirnagar University
Savar, Dhaka-1342
tajrean.shaharat@gmail.com

Mohammad Abu Yousuf
Institute of Information Technology
Jahangirnagar University
Savar, Dhaka-1342
yousuf@juniv.edu

***Abstract*— As a seventh most spoken native language, Bengali needs a robust and accurate optical character recognition (OCR) especially for number detection. As there is no publicly available well-organized dataset for Bangla number OCR, a synthesized dataset was generated to fill the lack of available data. The recent advancement in artificial intelligence using deep neural networks easily outperforms prior hand selected feature-based machine learning approaches. As the region proposal networks (RPN) in deep neural networks perform very well in detecting objects, it can be used for digit detection in an image. So, in this work a very robust Bengali handwritten number detection system is presented where with the help of deep neural networks and a very well-organized, unbiased generated dataset we achieved state of the art result in handwritten Bangla number detection. This system beats any related prior works by a large margin while considering a real world dataset for benchmarks. The overall detection accuracy was 97.8%. The processing can be done real-time with about 35 images per second using a GPU. Also, while implementing the solution is completely based on python, the framework used for deep learning is Google's Tensorflow and the dependencies, all of which are publicly available.**

I. INTRODUCTION

Bengali is the seventh most spoken native language in the world by population [1]. It's not only Bangladeshi national language but also the second most widely spoken language in India. So, every work regarding the digitalization and automation of the language directly affect about 208 million total speaker [2] who are using this language as their communication medium. There is a vast amount of data on handwritten text images in Bangla. If those data could be automatically digitalized via a good OCR, the information in many fields like medical data analysis, financial transaction analysis and automation in banks and other similar fields, automatic examination evaluation and result processing in academic sectors like schools, colleges etc. could be used by us. So, the automation in digitalization of those Bengali handwritten images in various sectors are becoming more and more essential and, in some cases a must have feature. With this increasing demand on automated system everywhere, an essential step is the automate the handwritten number detection in places like banks, offices, road signs etc. In Bengali number detection, many ways were tried to accurately detect the number from any image. But for the lack of unbiased dataset on Bengali number OCR and hand selected feature based machine learning approaches made the accuracy of Bengali number detection unusable in practical fields. Our solution makes use of a synthesized unbiased dataset for Bengali number detection which is NumtaDB [3] and then address the accuracy issue by implementing a deep learning based approach which uses Tensorflow [4] as the underlying framework. Used RPN based object detection for the digits detection because even though Recurrent Neural Network (RNN) is more efficient and accurate for character sequence detection but it requires larger real world dataset which is not available for Bangla numbers. That's why RNN based solutions exhibited poor accuracy. Also as we used FasterRCNN instead of RetinaNet (5) which has higher accuracy because FasterRCNN is relatively faster. And we didn't use SSD(6) object detection as it has too low accuracy. In the following sections we discuss elaborately about the existing solutions in **Section 2**, in **Section 3** we discuss the methodology of our solution, in **Section 4** we discuss the results and outputs of our system and finally in **Section 5** we conclude the works.

II. LETERATURE REVIEW

As the problem consists of multiple smaller parts and also can be solved in a number of ways. One approach is to segment each character from the input image and then classify each character to detect the number in the image. To do this we need to heavily depend on the accuracy of the segmentation also on the classification. But as we know that the handwritten Bengali numerals can be very hard to segment in a robust environment. As some hand selected features are not enough to segment the written characters accurately. The real-time accuracy in a robust dataset these hand selected feature based approaches show very poor accuracy and hence become unusable in practical usage. As this [7] paper shows an excellent way to segment connected characters in handwritten digits but the segmentation accuracy is not so high which can be shown to us, let alone that the demonstrated results are done in a very controlled environment and a private dataset [8, 9]. So the accuracy will suffer in real world data where the size, angle, position, depth of the character, lighting conditions etc. may affect the process very much [10-14]. Also for the other important part classification of the segmented characters most of the previous works on Bangla handwritten numerals uses biased which also can be shown to us and not so robust datasets like CMATERDB dataset [15], SUST-Bangla Handwritten Numeral Database (SUST-BHND), BanglaLekha-Isolated database etc. But very recently a relatively superior, well organized and unbiased dataset was published publicly called NumtaDB [3]. A better and robust system has been used in our solution. Also there are many recent works [16,17] to classify only a single character which is a major part of our

978-1-7281-5242-4/19 ©2019 IEEE

system. Many of those insights found in these works have been Incorporated to us.

III. Methodology

Our Handwritten Bangla Number detection system has multiplier steps involved in it. Firstly train a convolutional neural network to detect each digit location and the label of it from an image. To train the network, a dataset is generated from NumtaDB handwritten Bangla digits [3]. After training the network a serialized version of the network weights are saved as a model. The saved model is then used to detect the locations and the labels of the digits written in an image. Finally, the detected locations and labels are used to find the numbers. The steps are discussed by us in details below,

A. Requirement Specification

In this proposed method need to do, Detecting Bangla digits and number in noisy handwritten text image, varied size, user independent handwriting, noisy background, hard lighting condition etc.

B. Algorithm & Pseudocode

1. To Collect all 180 X 180 X 3 pixel single character images from NumtaDB data.

2. Then generate random sized blank images in different shades of gradients to simulate different paper color as the background.

3. On top of that use programmatically place the Numta digit images. Before randomly placing they next adjust the color so that the digit borders fuses with the color accent of the background.

4. The digit image was normalized with the background color so that the clone look natural.

5. To place a character randomly on the background the special factors are taken into consideration like the digits are neither plotted too densely nor do they have any overlaps.

6. After the dataset was generated they are trained using Tensorflow object detection API which.

7. By us used FasterRCNN for object detection and InceptionV2 architecture for as the underlying neural network.

8. Then trained the network for 20000 epoch.

9. After then freeze the training model to an inference model for better runtime optimizations.

10. Finally it has been detected the digits in any given image using the trained model.

11. Using some post processing like non overlapped detections and carefully tuned parameters for considering the detections as a single

12. Sample output in real world test

C. Generating Data

To train the neural network first a properly robust and unbiased data set is required. But there is no publicly available data set for handwritten Bangla numbers are available. To fill the unavailable data set for training a region proposal network based annotated dataset using NumtaDB data set [3] are generated which is publicly available, very well organized and carefully collected unbiased digit data set. Though this dataset only has the digits and their hand annotated labels of those digits (Sample images is shown in Fig. 1), for our work this dataset is not properly fit to use directly.

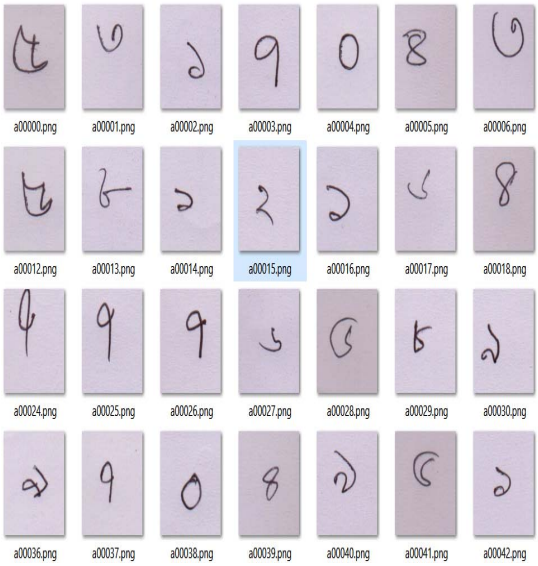


Fig. 1. Sample images from NumtaDB

A proper RPN based dataset is generated from those data found in NumtaDB. The data generation process is briefly described in the flowchart shown in Fig. 2:

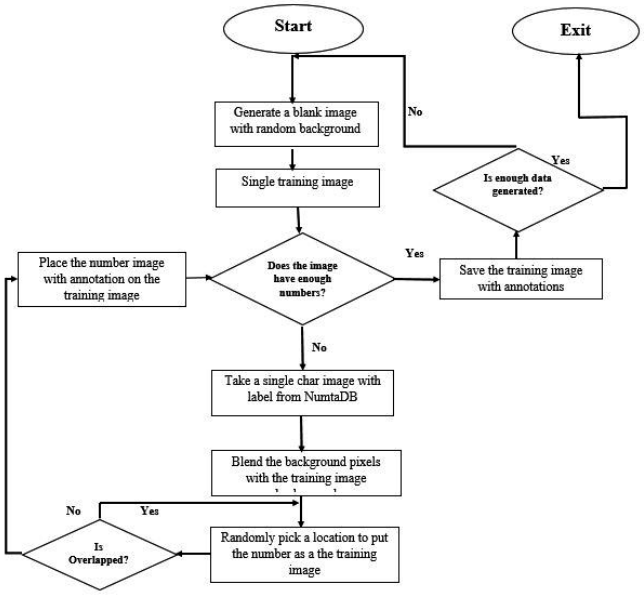
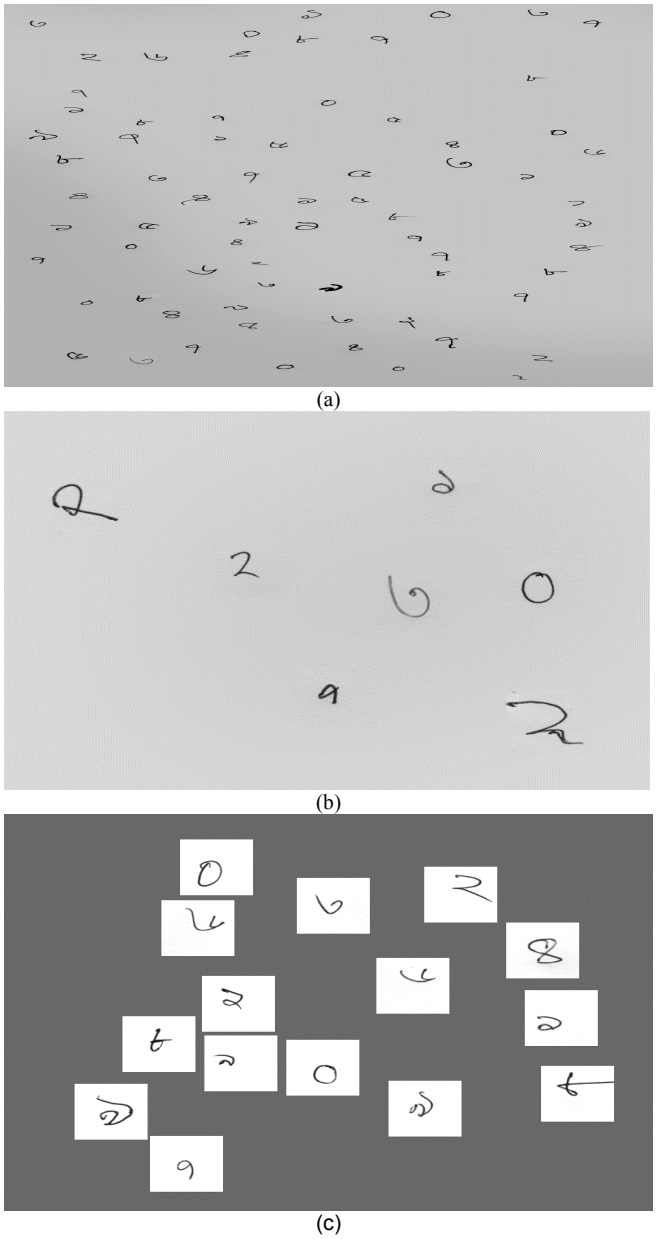


Fig. 2. Data generation flowchart

To generate a single image with annotations a blank random resolution image is created with random background color. Then the blank image is filled with single digit images found in NumtaDB and placed randomly to simulate a page with handwritten digits in it. As the background of each single digit image from NumtaDB is white, to match the background with the synthesized annotation image background the background pixels are replaced with the appropriate color so that the image looks natural and looks like a real handwritten image, see Fig. 3(**a** and **b**). If the

background color was not corrected the generated image would look like the image on Fig. 3(**c**).



(a)

(b)

(c)

Fig. 3. (a) Real Sample generated images for training **A.** (b) Sample generated images for training **B** (c) **C** is for demonstration purposes only to show how it would look like if the background color was not normalized.

D. Convolutional Neural Network for Region Proposal Network

Convolutional networks are at the core of most state of the art computer vision solutions for a variety of tasks. Other hand selected feature based machine learning approaches are easily outperformed by the deep convolutional networks. In this work, we used Faster RCNN [18] for region proposal network and also for classification of the detected digit on the image. Using the recently popular terminology of neural networks with 'attention' mechanisms, the RPN module tells the Fast R-CNN module where to look. So the accuracy of the detection is much higher while retaining the processing time reasonably well. Using this object detection based approach, we convert the character classification problem to an object detection OCR solution also use inception v2 [19] as the backbone network. The RPN [2o] work in 3 steps.

1. In the first step, the input image goes through a convolution network which will output a set of convolutional feature maps on the last convolutional layer:

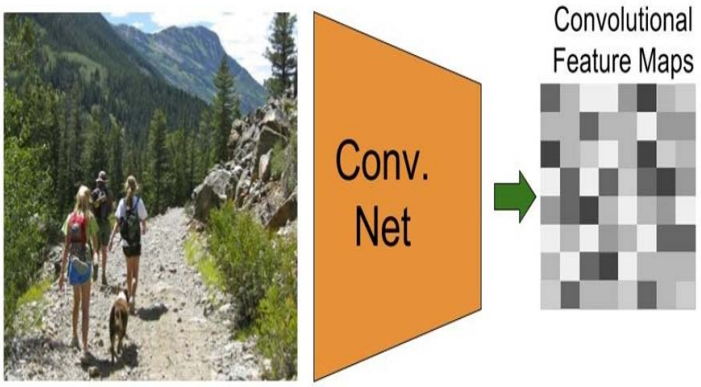


Fig. 4. Convolutional Feature Maps [20]

2. Then a sliding window is run spatially on these feature maps. The size of sliding window is n×n (here 3×3). For each sliding window, a set of 9 anchors are generated which all have the same center (xa,ya) but with 3 different aspect ratios and 3 different scales as shown below. Note that all these coordinates are computed with respect to the original image

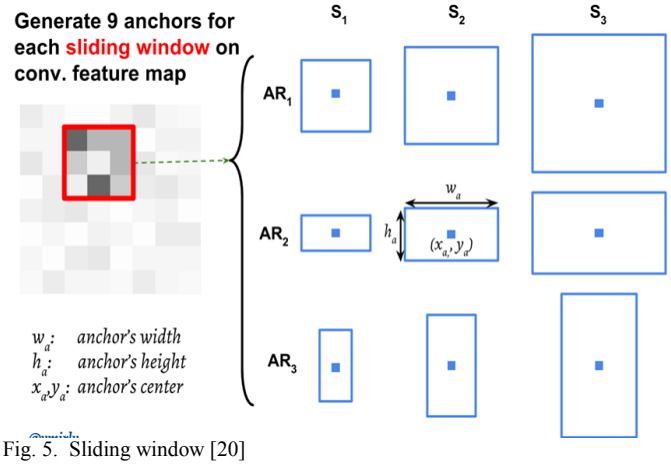


Fig. 5. Sliding window [20]

Furthermore, for each of these anchors, a value p∗ is computed which indicated how much these anchors overlap with the ground-truth bounding boxes.

p* = {1 if IoU > 0.7, -1 if Iou < 0.3 otherwise 0

Where IoU is intersection over union and is defined below:

IoU = Anchor∩GTBox / Anchor∪GTBox

3. Finally, the 3×33×3 spatial features extracted from those convolution feature maps (shown above within red box) are fed to a smaller network which has two tasks: classification (cls) and regression (reg). The output of regressor determines a predicted bounding-box (x,y,w,h)(x,y,w,h),

The output of classification sub-network is a probability p indicating whether the predicted box contains an object (1) or it is from background (0 for no object).

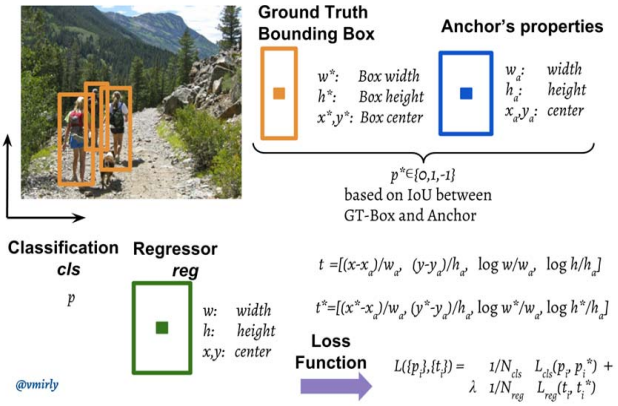


Fig. 6. The 3×33×3 spatial features [20]

E. *Training RPN*

The RPN can be trained end-to-end by back propagation and stochastic gradient descent (SGD) [21]. We follow the "image-centric" sampling strategy from [22] to train this network. Each mini-batch arises from a single image that has many positive and negative example anchors. It is possible to optimize for the loss functions of all anchors, but this will bias towards negative samples as they are dominant. Instead, it randomly sample 256 anchors in an image to compute the loss function of a mini-batch, where the sampled positive and negative anchors have a ratio of up to 1:1. If there are fewer than 128 positive samples in an image, it pad the mini-batch with negative ones. Training process is briefly shown in Figure 7 flowchart below:

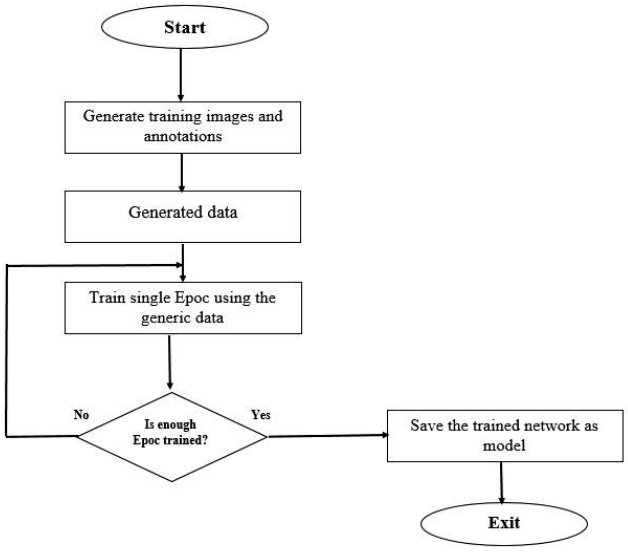


Fig. 7. Training RPN process flowchart

The generated annotation data discussed earlier, is used to train the network. Total number of training Epoc is **200000**, about **1000 images** are used as training set and **200 images** is used for validation set. Initial learning rate is 0.0002 then progressively decreased. The learning rate curve is show in Fig 8:

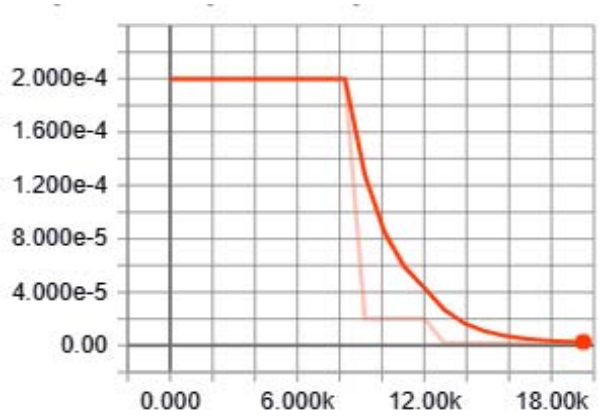


Fig. 8. Learning rate curve for training.

There have different loss curves during training the network. The different loss curves is shown in Fig. 9:

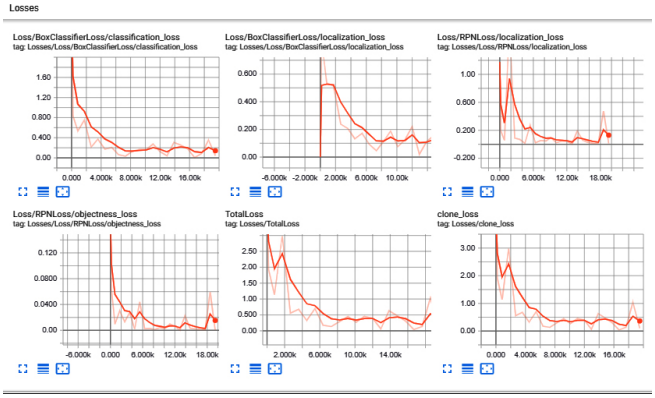


Fig. 9. Different loss curves

Serial wise there have total six different loss curves

1. BoxClassifierLoss/classification_lose,
2. Clone loss,
3. BoxClassifierLoss/Localization_loss,
4. RPNLoss/Objectness_loss
5. RPNLoss/localization_loss,
6. Total_loss

After training the network for 200000 Epocs the weights for the network is serialized and save as a model for future inferences. This saved model can be used to detect any image.

F. *Postprocessing For Number Detection*

After training the neural network the model can successfully predict the digit labels and the location of the digit in an arbitrary image. These two information for every digit is then used to find the correct number formed by those digits in an image. To do that the location information is used to recognize the alignment of each digits. Then the digits are grouped together by using the Euclidian distance between the digits. Following equation 1 shows that if the location of a detection is (x, y) and another one is (a, b) in Cartesian coordinate system the distance is found via equation 1.

$$\text{Dist}((x, y), (a, b)) = \sqrt{(x - a)^2 + (y - b)^2} \quad (1)$$

If the distance goes beyond double the width of that digit it can be considered that the digit is of another number. Other than that it can be safely assumed that the digits are from one single number and together they form the whole number. Also, the height and size of the digit is taken into consideration too. As a single image can have different scale digits and numbers written in it the system needs to correctly identify the difference among those different sized

numbers. If the area of the detected digits is 50% larger than the other digits it is considered to be on other scale. So, different numbers. An example can be found about this different scaled numbers in Fig. 10.The confusion matrix for only the classification on NumtaDB dataset it shown below,

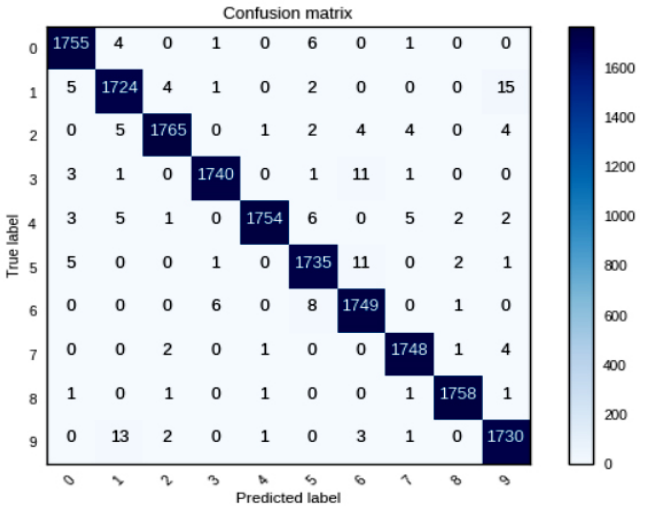
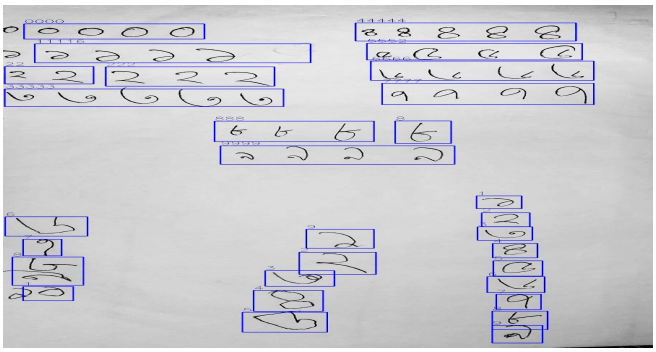


Fig. 10. Confusion matrix on digit classification

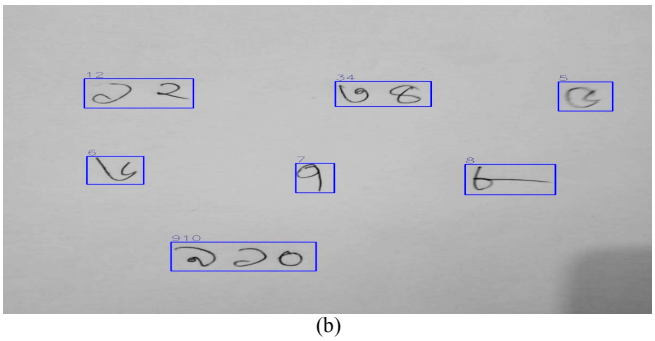
As it is clearly shown that the classification accuracy is very consistent for all the digits individually. It is known that the two digits in Bengali numerals '1' and '9' are very similar visually. But the classification accuracy is very high regardless of those similar looking digits in the dataset. As the superior neural networks were used for feature detection and recognition and also via using the data augmentations the accuracy on digit classification is very impressive and practical to use in real life applications And for the localization part of the digits, the features from the neural networks are used here too, to make the detection process more optimized for resource consumptions and to increase the accuracy of the detection. As the neural network learns from the training data that both the classification and the localization both features need to selected from the same training dataset. The overall accuracy is very consistent in complex scenarios like varied background, illumination change, noisy background, different scale of written digits etc.

IV. Result

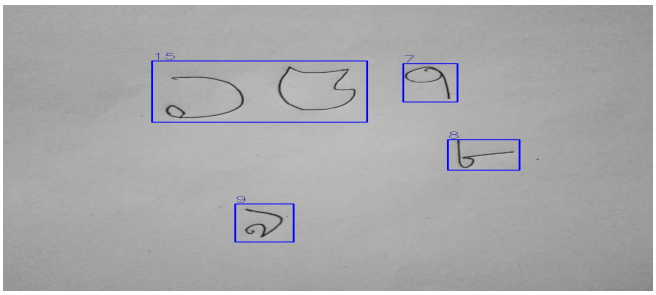
The proposed method on Handwritten Bengali Number Detection using Region Proposal Network. It's very helpful specially for foreigners, who can't read Bangla digit or number. Using for this application there have some sample images are shown in fig 8: The images used in this benchmark are completely unbiased and was not used in training the network. It's not only detect Bengali digit, it can also detect Bengali number. The trained network can detect handwritten Bangla numbers in all the different lighting conditions and samples written by any random persons. As we've used neural network for RPN and classification of the digits, the detection accuracy was consistent and robust. During training the convolutional network for prediction RPN was converged at around 18000 Epoc. At this point the total loss was 0.203 which is down from initial 15.327. The minimum total loss was near 0.195 at some point but as at that point it was not the highest overall accuracy considering both the classification and localization accuracy.



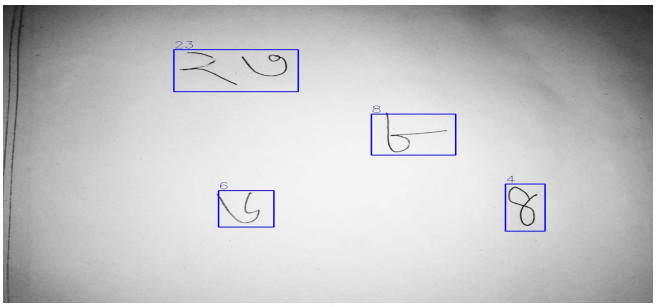
(a)



(b)



(c)



(d)

Fig. 11. (a) (b) (c) (d) Sample output images

During training the convolutional network for prediction RPN was converged at around 18000 Epoc. At this point the total loss was 0.203 which is down from initial 15.327. The minimum total loss was near 0.195 at some point but as at that point it was not the highest overall accuracy considering both the classification and localization accuracy. From the confusion matrix it is clear that the classification accuracy is very high. But if we look deeply into the error cases in classification step, the most errors are between classes 1 and 9. Which are very similar to see and they can be confusing even to the human eye in some cases. As sometimes the smaller resolution cases few pixels to misrepresent the actual information the network gets confused with those class, yet the error rate is very low. Only 15 1's out of about 1800, was miss classified as 9's. Similarly we have 3 and 6

in Bengali are also such confusing case. Where in some cases the starting pixels at the bend of character 6 can be misunderstood as being character 3 in some cases. This error rate is even lower. Only 6 error cases were reported in the confusion matrix. Also, in localization of the characters we have seen the error rate is quite low and thus the result is very high. As Faster RCNN algorithm was used for object detection on top of inception version two backbone network, the accuracy of the RPN prediction was very high. We have got more than 45 mAP (Mean Average Precision) which is quite satisfactory result for such small object. The solution was tested on about 200 images of Bangla handwritten images. The overall detection accuracy was 97.8%. The RPN accuracy was 98.43% and the classification accuracy was 99.35%. The processing time is optimized after training the model to detect in real time. The GPU uses was "Nvidia Gforce GTX 1080 TI". This solution can detect handwritten Bangla numbers with about 35 images per second using a GPU. Also the processing time using CPU is also about 10 images per second.

V. Conclusion

In this research, main investigation is the performance of several popular deep convolutional neural networks for hand written Bangla number recognition. So it is a demand of time to build a highly efficient Bangla number OCR. A good Bangla number OCR many help us in many ways life using information of a check, receipt, hand note etc. It will save our valuable time as well as resource. This system will have more than 97.8% accuracy after a good amount of training. Also the detection time in CPU and GPU is much optimized to use in real world applications. RPN is the one true backbone and have proven to be very efficient till now. Its purpose is to propose multiple objects that are identifiable within a particular image. That's why RPN used for Handwritten Recognition Number detection. To the best of knowledge, this is the best recognition results for Bangla handwritten number OCR. The previous works in this fields are heavily based on the hand selected feature based machine learning approaches. This work greatly outperforms any previous works in this field. As deep learning comes into play when we consider robust feature selection and overcoming the detection problems generated from the different lighting conditions, illuminations, exposure, handwriting from different styles and persons etc. In near future, this field of handwritten Bangla number OCR will overcome some shortcomings. For example, some fusion based models will be explored and developed for handwritten Bangla Number recognition and also make this Bangla number OCR more accurate.

References

1. "The world factbook, "https://www.cia.gov/library/publications/the-world-factbook/geos/xx.html, accessed: 2018-05-26
2. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," Neural computation, 1989.
3. "Ethonologue," https://www.ethnologue.com/19/language/ben/,accessed: 2018-05-26
4. Alam S, Reasat T, Doha RM, Humayun AI, "NumtaDB-Assembled Bengali Handwritten Digits," arXiv preprint arXiv: 1806.02452, 2018.
5. Mart´ın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Man´e, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Vi´egas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org
6. "Focal Loss for Dense Object Detection" https://arxiv.org/abs/1708.02002, accessed: 2019-02-01.
7. "Single Shot MultiBox Detector" https://arxiv.org/abs/1512.02325, accessed: 2019-02-01.
8. Md. Aktaruzzaman, Md. Farukuzzaman Khan, Dr. Ahsan-Ul-Ambia, "A New Technique for Segmentation of Handwritten Numerical Strings of Bangla Language," IJITCS, 2013
9. Subhadip Basu, Nibaran Das, Ram Sarkar, Mahantapas Kundu, Mita Nasipuri, Dipak Kumar Basu, "An MLP based Approach for Recognition of Handwritten 'Bangla' Numerals," ICERIE 2017, SUST
10. Md. Mahbubar Rahman, M. A. H. Akhand, Shahidul Islam, Pintu Chandra Shill, "Bangla Handwritten Character Recognition using Convolutional Neural Network," ijigsp, 2015.
11. Aneel KUMAR Chintha, "Handwritten Digit Recognition Using Structural, Statistical Features and Knearest Neighbor Classifier," IJIEEB, 2014
12. Haider Adnan Khan, "Handwritten Bangla Numeral Recognition using Local Binary Pattern," ICEEICT, 2015.
13. U. Pal, B. B. Chaudhuri, "OCR in Bangla: An Indo-Bangladeshi Language", Proc. of 12th Int. Conf. on Pattern Recognition, IEEE Computer Society Press, pp. 269-274, 1994.
14. U. Pal, T. Wakabayashi, and F. Kimura, "Handwritten Bangla Compound Character Recognition Using Gradient Feature," in 10th International Conference on Information Technology (ICIT 2007), pp. 208-213, 2007.
15. H. A. Khan, A. Al Helal, and K. I. Ahmed, "Handwritten Bangla digit recognition using Sparse Representation Classifier," in 2014 International Conference
16. N. Das, R. Sarkar, S. Basu, M. Kundu, M. Nasipuri, and D. K. Basu, "A genetic algorithm based region sampling for selection of local features in handwritten digit recognition application," Applied Soft Computing, vol. 12, no. 5, pp. 1592–1606, 2012.
17. Ashadullah Shawon, Md. Jamil-Ur Rahman, Firoz Mahmud, M.M Arefin Zaman, "Bangla Handwritten Digit Recognition Using Deep CNN for Large and Unbiased Dataset," ICBSLP, 2018.
18. Mamunur Rahman Mamun, Zabir Al Nazi, Md Salah Uddin Yusuf, "Bangla Handwritten Digit Recognition Approach with an Ensemble of Deep Residual Networks," International Conference on Bangla Speech and Language Processing (ICBSLP), 2018.
19. Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," arXiv:1506.01497, 2015
20. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, "Rethinking the Inception Architecture for Computer Vision," arXiv:1512.00567, 2015.
21. "How does the region proposal network (RPN) in Faster R-CNN work?" https://www.quora.com/How-does-the-region-proposal-network-RPN-in-Faster-R-CNN-work., accessed: 2019-02-01.
22. R. Girshick, "Fast R-CNN," in IEEE International Conference on Computer Vision (ICCV), 2015.

An Annotated Bangla Sentiment Analysis Corpus

Fuad Rahman
Apurba Technologies Ltd.
Dhaka, Bangladesh
fuad@apurbatech.com

Habibur Khan
Apurba Technologies Ltd.
Dhaka, Bangladesh
habib@apurbatech.com

Zakir Hossain
Apurba Technologies Ltd.
Dhaka, Bangladesh
zakir@apurbatech.com

Mahfuza Begum
Apurba Technologies Ltd.
Dhaka, Bangladesh
mahfuza@apurbatech.com

Sadia Mahanaz
Apurba Technologies Ltd.
Dhaka, Bangladesh
sadia@apurbatech.com

Ashraful Islam
Apurba Technologies Ltd.
Dhaka, Bangladesh
ashraful@apurbatech.com

Aminul Islam
Apurba Technologies Ltd.
Dhaka, Bangladesh
aminul@apurbatech.com

978-1-7281-5242-4/19

***Abstract* – This paper presents a Bangla corpus specifically targeted for sentiment analysis and made available to researchers under an open-source licensing scheme[1]. We have collected and manually annotated over 10,000 sentences with sentiment polarity. We then moved to the Word domain and annotated over 15,000 words derived from these sentences with sentiment polarity. Each entry is the corpus has been cross-annotated by at least two and sometimes three annotators for ensuring quality. Also as a pre-requisite of creating a high quality sentiment analysis corpus, we had to build a secondary corpus for Bangla word stemming, which is also been cross-validated by at least two and sometimes three annotators for ensuring quality.**

Index Terms – Sentiment Analysis, NLP, Bangla Corpus, Annotation, Open Source Corpus

I. Introduction

Sentiment analysis is a very important part of natural language processing. While very robust solutions for English already exists both in academic and commercial domains, for Bangla language, work in this area is still in its infancy. As the focus of tools for sentiment analysis has now shifted from rule based to machine learning methods, the need for annotated and ground truth data for training these solutions are of utmost importance. Unfortunately there is almost no serious corpus for Bangla language that is available for sentiment analysis, forcing researchers to stich together their own small corpora which are neither standardized and nor rigorously quality controlled. In this paper, we present a fully annotated corpus for sentiment analysis.

A. Brief Background

In recent times, there has been some research reported on Bangla sentiment analysis. One common approach seems to be to translate a Bangla word and then use the polarity from the English translated word. [1][6][7]. While it works on straightforward words, it cannot handle the nuances of a language. For example the word "জটিল" means "complex" [2] and it has a negative sense. But in Bangla it is often used in a positive sense, for example, "তামিম আজ জটিল খেলছে". Another example is the word "খাওয়া" which is commonly translated to "eat", the common polarity of which is neutral. But in the sentence "তার খাওয়া নাই", the polarity is distinctly negative.

The reason for the popularity of this type of approach is very simple, a distinct lack of a ground truth corpus suited for training machine learning algorithms. Although there are some existing data set for sentiment analysis, but most of these are not available publicly. Some publicly available data set are small, e.g. [4] has about 4,000 sentences, whereas [1] has about 7,000 sentences.

One of the most significant resources is described in [6]. This corpus size is around 10,000 sentences. These sentences were collected from Facebook, Twitter, YouTube, online news portals and product review pages.

[1] See Section VI for details

II. Sentiment Analysis Corpus

A. Data Source

The source of this data is the online Sports section of Prothom Alo, as shown in Fig 1 below.

Fig. 1 Daily Prothom Alo Online Edition.

Most of source sentences for the corpus were collected from the comments Section as shown in Fig 2 below.

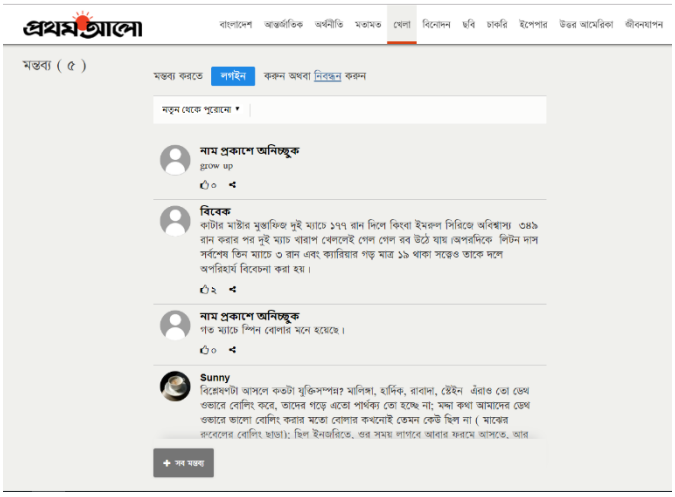


Fig. 2 Comments in the Sports Section.

B. Methodology

The corpus was prepared in a combination of manual and automated steps. Initially the sentences appearing in the Sports Section were copied manually. These data were labeled by hand into three categories "positive", "negative" and "neutral", by a "Content Team" and then crosschecked by a "QA Team".

The truth labeling is done at two levels.

Sentence Level: The first level is the sentence level, as seen in Fig 3 below. In this case, the polarity applies to the overall sentence.

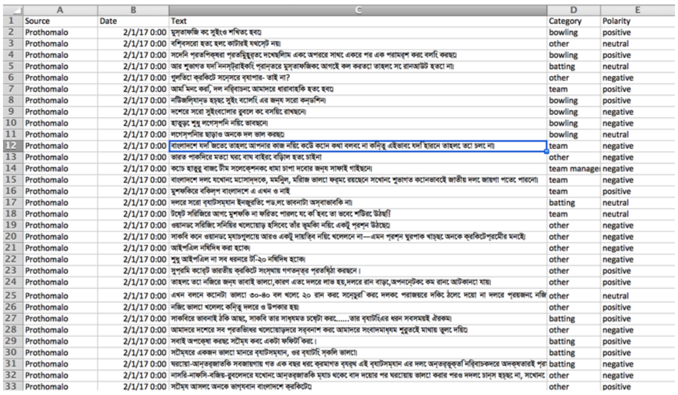


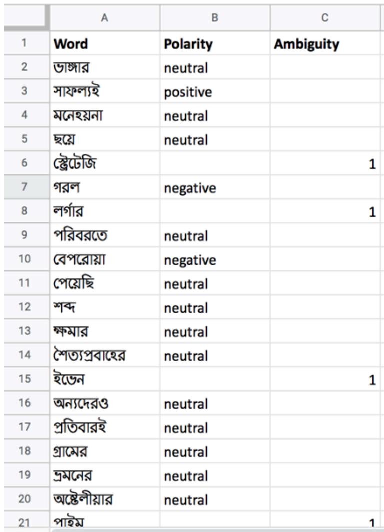
Fig. 3 Sentence level polarity.

- Manual collection and preprocessing: We collected more than 10,000 sentences from comments from online Bangla newspaper (https://www.prothomalo.com/), primarily from cricket sports news. We only included valid and complete single sentences. The task was distributed among the 5 members team members, who are all native Bangla speakers. Each member tagged sentiment polarity for the sentences allocated to him/her. Another member then crosschecked this. A third member then re-assigned the polarity sentences if the first and second members disagreed. The same methodology was applied to crosschecking sentence validation. In case there are disagreements, the assigned final polarity is at least assigned by two team members. If all three team members disagreed, the sentence was considered to be too ambiguous and dropped from the corpus.
- Automatic processing: We removed unwanted characters, words and symbols from the sentences, such as:
    - ['১','২','৩','৪','৫','৬','৭','৮','৯','1','2','3','4','5','6','7','8','9']
    - ['□','?',',','.',';',':','.','(',',-',',–',',/',',_','*','%','!','\'','+','<','>','—','০','=']
    - ['"','"','|',',…',')','`','@','#','।','"','&','–','_','😏','🗣','👋','😀','😊']
    - [A-Z]
    - [a-z]

  We also removed duplicate sentences.

Word Level: The second is polarity on the word level, as shown in Fig 4 below.

- Automatic processing: We tokenized all collected sentences and removed numbers, digits, and symbols from the tokenized word list. We then identified the unique words from the word list of nearly 15,000 words. We stemmed the word list using two different stemmers, StemmerR[10] and StemmerP[11]. We then identified the words that produced the same result from these two stemmers.

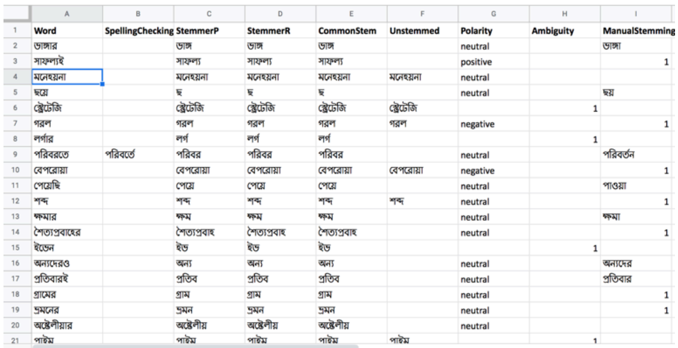


	A	B	C
1	Word	Polarity	Ambiguity
2	ভাঙ্গার	neutral	
3	সাফল্যই	positive	
4	মনেহয়না	neutral	
5	ছয়ে	neutral	
6	স্ট্রেটেজি		1
7	গরল	negative	
8	লর্গার		1
9	পরিবরতে	neutral	
10	বেপরোয়া	negative	
11	পেয়েছি	neutral	
12	শব্দ	neutral	
13	ক্ষমার	neutral	
14	শৈত্যপ্রবাহের	neutral	
15	ইডেন		1
16	অন্যদেরও	neutral	
17	প্রতিবারই	neutral	
18	গ্রামের	neutral	
19	ভ্রমনের	neutral	
20	অস্ট্রেলীয়ার	neutral	
21	পাইম		1

Fig. 4 Word level polarity.

- Manual collection and preprocessing: We checked whether the un-stemmed word is already a root or not and manually corrected the roots for those words that were stemmed wrongly. Once a clean word list was created, we then tagged the polarity of each word manually, using the same three-tiered approach as described before. This step also resulted in identifying some words that were ambiguous. These are then dropped for the final corpus.

A snapshot of this corpus is shows in Fig 5 below.



	A	B	C	D	E	F	G	H	I
1	Word	SpellingChecking	StemmerP	StemmerR	CommonStem	Unstemmed	Polarity	Ambiguity	ManualStemming
2	ভাঙ্গার		ভাঙ্গ	ভাঙ্গ	ভাঙ্গ		neutral		ভাঙ্গা
3	সাফল্যই		সাফল্য	সাফল্য	সাফল্য		positive		1
4	মনেহয়না		মনেহয়না	মনেহয়না	মনেহয়না	মনেহয়না	neutral		
5	ছয়ে		ছ	ছ	ছ		neutral		ছয়
6	স্ট্রেটেজি		স্ট্রেটেজি	স্ট্রেটেজি	স্ট্রেটেজি	স্ট্রেটেজি		1	
7	গরল		গরল	গরল	গরল	গরল	negative		1
8	লর্গার		লর্গ	লর্গ	লর্গ			1	
9	পরিবরতে	পরিবর্তে	পরিবর	পরিবর	পরিবর		neutral		পরিবর্তন
10	বেপরোয়া		বেপরোয়া	বেপরোয়া	বেপরোয়া	বেপরোয়া	negative		1
11	পেয়েছি		পেয়ে	পেয়ে	পেয়ে		neutral		পাওয়া
12	শব্দ		শব্দ	শব্দ	শব্দ	শব্দ	neutral		1
13	ক্ষমার		ক্ষম	ক্ষম	ক্ষম		neutral		ক্ষমা
14	শৈত্যপ্রবাহের		শৈত্যপ্রবাহ	শৈত্যপ্রবাহ	শৈত্যপ্রবাহ		neutral		1
15	ইডেন		ইড	ইড	ইড			1	
16	অন্যদেরও		অন্য	অন্য	অন্য		neutral		অন্যদের
17	প্রতিবারই		প্রতিব	প্রতিব	প্রতিব		neutral		প্রতিবার
18	গ্রামের		গ্রাম	গ্রাম	গ্রাম		neutral		1
19	ভ্রমনের		ভ্রমন	ভ্রমন	ভ্রমন		neutral		1
20	অস্ট্রেলীয়ার		অস্ট্রেলীয়	অস্ট্রেলীয়	অস্ট্রেলীয়		neutral		
21	পাইম		পাইম	পাইম	পাইম	পাইম		1	

Fig. 5 Corpus after stemming using two different algorithms.

III. Corpus Statistics

This Section qualifies the corpus.

TABLE I
RAW DATA COLLECTED FOR THE SENTIMENT CORPUS

Total number of sentences	10,008
Total number of words before filtering	19,731

Total number of words after filtering	14,874
How many ambiguous	2,140
How many words accepted	12,734

Table 1 shows the statistics of raw data collected from the sentiment corpus.

TABLE 2
COMPARING THE TWO STEMMERS

	StemmerP	StemmerR
How many words were stemmed	14,874	14,874
How many times the stemmer was able to stem a word	14,662	14,874
How many times the stemmer was not able to stem a word	212	14,874

Table 2 compares the performance of the two stemmers. This step was completely manual and three-level peer-viewed.

We found that 1.5% times the stemmers did not agree with each other. In order to correct that, we manually fixed the stemming.

TABLE 3
MANUAL STEMMING

Category	Number
Accuracy of StemmerP	58.08%
Accuracy of StemmerR	59.65%
How many words were manually stemmed	5,138
How many spelling were corrected manually	682

Table 3 shows the results of manually fixing stemming issues.

After all this cleanup and manual fixes, the final corpus has the following entries, as seen in Table 4.

TABLE I
RAW DATA COLLECTED FOR THE SENTIMENT CORPUS

	Positive		Negative		Neutral	
	Actual	%	Actual	%	Actual	%
Total Number of sentences	3,183	33.0	4,110	42.67	2,337	24.27
Total Number of words	824	6.47	1,068	8.38	10,804	84.80

Fig. 6 WordCloud of corpus sentences with both positive and negative polarity.

IV. CORPUS PROPERTIES

A. *WordCloud of Collected Sentences*

Fig. 6 shows a WordCloud of the collected sentences with positive and negative polarity. Please note that we have used a stop word list to filter the words before this was created. We have identified 398 stop words. The same applies to building WordClouds for negative and neutral sentences.

Fig. 7 shows a WordCloud of the collected sentences with positive polarity.

Fig. 7 WordCloud of corpus sentences with a positive polarity.

Fig. 8 shows a WordCloud of the collected sentences with negative polarity.

Fig. 8 WordCloud of corpus sentences with a negative polarity.

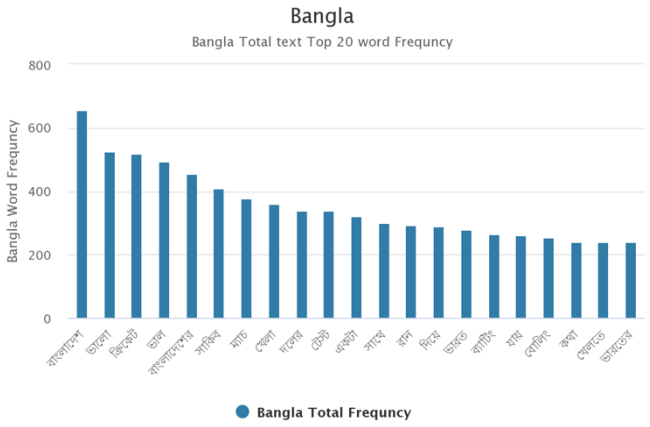


Fig. 9 Word frequency of top 20 words

B. Frequency Distribution of Top 20 Words

Fig. 9 shows the frequency of the top 20 words in our corpus.

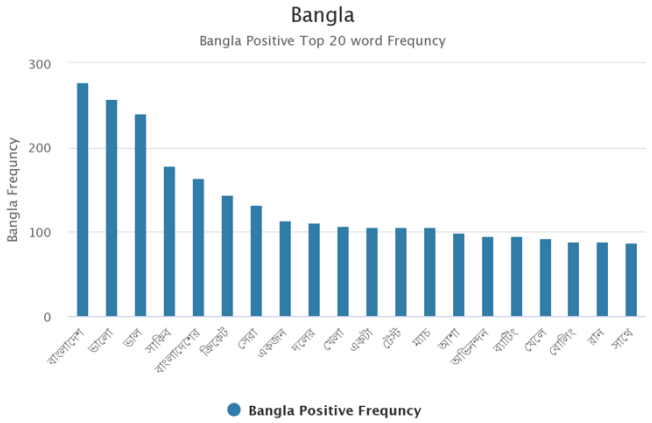


Fig. 10 Word frequency of top 20 positive words

Fig. 10 shows the frequency of the top 20 positive words in our corpus.

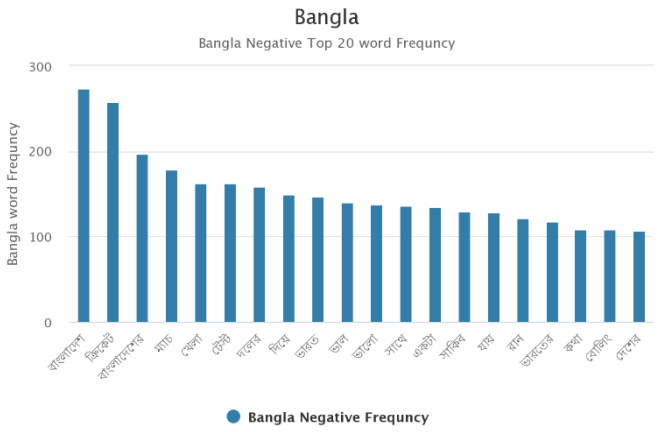


Fig. 11 Word frequency of top 20 negative words

Fig. 11 shows the frequency of the top 20 negative words in our corpus.

V. Some Observations

We have created this corpus from a very focused source, the sports news domain and have incorporated sentences, unique words and stemmed words. We extensively cleaned the data using a combination of filtering and stop word list — employing both manual and automated process. Every entry is cross-validated using at least two, and sometimes three annotators. We have manually corrected misspelling and stemming errors. So this is not just an annotated and ground truth corpus on sentiment analysis, it is also a corpus for training stemming engines.

It was also very important for us to build in auditability in the corpus. That is why every word and root word is cross-referenced against the source sentences. This way this corpus can be adopted for other NLP solutions with ease.

The other aspect of our corpus design is the transparency of the data collection process. It is a natural extension of the auditability of the data mentioned above.

VI. Open Source Licensing

Not-For-Profit and academic organizations and government agencies may use this corpus for noncommercial linguistic research and education only. For-profit organizations may use this corpus after signing a commercial technology development contract. Not-For-Profit and academic organizations and government agencies cannot use this corpus to develop or test products for commercialization, nor can they use this in any commercial product or for any commercial purpose.

VII. Conclusions and Future Work

We have presented a Bangla corpus specifically targeted for sentiment analysis. We described the methodology, source and clean up process. In the future, we plan to extend the corpus to support aspect based sentiment analysis for Bangla, where different clauses of a single sentence may have different sentiments. We also plan to extend this by adding sentiments for phrases, idioms and clauses. In addition, we plan to offer a set of machine learning models that can use this corpus.

References

[1] Adrija Roy and Abhishek Anand Singh. Sentimental Analysis (Bengali) https://github.com/abhie19/Sentiment-Analysis-Bangla-Language.

[2] English & Bengali Online Dictionary & Grammar. http://www.english-bangla.com/bntoen/index/%E0%A6%9C%E0%A6%9F%E0%A6%BF%E0%A6%B2.

[3] Tazim Hoque. Word and Doc2Vec file for Bengali Sentiment Analysis. https://www.kaggle.com/tazimhoque/bengali-sentiment-text/.

[4] Atik Rahman. Bangla Aspect Based Sentiment Analysis Dataset. https://github.com/AtikRahman/Bangla_ABSA_Datasets.

[5] Md. Atikur Rahman and Emon Kumar Dey. Datasets for Aspect-Based Sentiment Analysis in Bangla and its Baseline Evaluation. 4 May 2018. Institute of Information Technology, University of Dhaka, Dhaka 1000, Bangladesh. https://res.mdpi.com/data/data-03-00015/article_deploy/data-03-00015.pdf?filename=&attachment=1

[6] Asif Hassan, Mohammad Rashedul Amin, Abul Kalam Al Azada, Nabeel Mohammed. Sentiment Analysis on Bangla and Romanized Bangla Text (BRBT) using Deep Recurrent models. 24 Nov 2016. Dept. of Computer Science and Engineering University of Liberal Arts Bangladesh (ULAB), Bangladesh. https://docs.google.com/viewerng/viewer?url=http://resources.apurbatech.com/publication/upload/1610.00369.pdf

[7] D. Das and S. Bandyopadhyay, Developing Bengali WordNet Affect for Analyzing Emotion. Int. Conf. on the Computer. Processing of Oriental Languages, pp. 35-40, 2010.

[8] Cliff Goddard. Natural Language Processing, Edition: 2nd edition, Chapter: 5, Publisher: CRC Press, Taylor & Francis, Editors: Nitin Indurkhya, Fred J. Damerau, pp.92-120

[9] Mohammad Daniul Huq. Semantic values in Translating from English to Bangla, The Dhaka University Journal of Linguistics: Vol. 1 No.2 August, 2008 Pages: 45-66.

[10] Rafi Kamal. Bangla Stemmer. https://github.com/rafi-kamal/Bangla-Stemmer.

[11] Sazedul Islam. Rule based Bengali Stemmer written in python. https://pypi.org/project/py-bangla-stemmer/.

Synthetic Class Specific Bangla Handwritten Character Generation Using Conditional Generative Adversarial Networks

Zinnia Khan Nishat
Department of Computer
Science and Engineering
University of Asia Pacific
Email: zinniakhannishat@gmail.com

Md Shopon
Department of Computer
Science and Engineering
University of Asia Pacific
Email: shopon@uap-bd.edu

***Abstract*—Bangla handwritten character recognition is known to be one of the most classical problem in the field of machine learning. In order to solve a machine learning problem one must thing is dataset. The more varied data a model sees the better it learns. Generative adversarial networks (GANs) are a group of neural networks that are used in unsupervised machine learning. It helps to resolve many difficult operations such as image generation from description, transforming low resolution image into high resolution, retrieving image contents given a small pattern etc. GAN's have many other promising applications in machine learning. There are many variations available for GAN. One of the variation of GAN is Conditional Generative Adversarial Networks(cGAN). This kind of GAN is used for generating a specific type of image. In this work we have used cGAN for generating Class Based Character Generation. This work can help researchers to generate handwritten characters to enhance the perfomance of deep learning models. We have trained this model to generate 50 Basic Bangla Characters, 10 Bangla Numerals and 24 Compound characters.**

***Index Terms*—Artificial Neural Network; Genrative adversarial networks; Convolutional Neural Network; Deep Learning; Handwritten Character Generation; Adversarial Models**

I. INTRODUCTION

Most of the known successes in the area of neural networks and deep learning has came from the application of neural network models to supervised machine learning tasks. The most visible of them are convolutional neural networks(CNNs)[1], recurrent neural networks(RNNs).[2] This models are nowadays achieving excellent results on different tasks such as classification[3], machine translation[4] and many more. But despite having so much success, a major obstacle in supervised learning is enough data to learn features that learns better structure of the data. Without having enough labeled data supervised learning fails to give its best. Labeling datasets are very time and cost consuming. But this is where unsupervised machine learning comes. Unsupervised machine learning is a type of learning that is used for drawing conclusion from datasets comprising of input data besides any labels.Unsupervised machine learning is aligned with what we cal true artificial intelligence. It is the idea that a computer can learn and understand to classify patterns and images barring a human to provide proper guidance.

978-1-7281-5242-4/19

Generative adversarial networks (GANs) are a kind of neural networks that are applied in unsupervised learning. GANs are mostly popular for image generation, generating high resolution image from low resolution, generating 3D objects and video frame predicting. In 2014, Ian Goodfellow and his colleague introduced GAN first.[5] After that it gained a large interest among researchers in recent years. AI pioneer Yann LeCun has called GANs the most fascinating idea in the last 10 years in machine learning. Using a combination of game theory and computational graph they showed that, two neural network models fighting with each other would be capable to co-train using plain old backpropagation. GAN is accomplished by a system of two neural network model, first one is Generator and second one is called Discriminator.

GANs are a branch of unsupervised machine learning, accomplished by a system of two neural networks, named as Generator and Discriminator. The two networks play a minimax game with each other. The generator tries to yield fake images in order to fool the discriminator into thinking that they are original image and the discriminator will try to discriminate among the real and the generated image as good as it can. They both keep playing this game until the discriminator becomes unable to discriminate/distinguish between the real and the fake image anymore.

In this paper, we will be using a special kind of generative adversarial networks in order to generate Bangla handwritten characters. We will be using a variation of GAN which is known as conditional generative adversarial networks.

II. BACKGROUND

Image generation is a common task in the field of neural network. Handwritten character generation is also a common task but handwritten Bangla character generation is completely a novel work. As already mentioned in this paper we will be using GAN to generate specific Bangla characters. GANs consist of two neural network models. We will take a look at some terminologies that are related to GAN and some of the previous work that were done on our task domain.

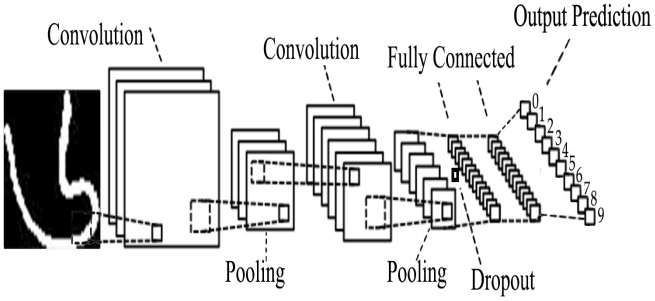


Fig. 1: Architecture of Convolutional Neural Networks

A. Previous Works

We consider our work a novel work. As of our knowledge there haven't been any work done which are class based handwriten character generation. Sikder et al[6] used vanilla generative adversarial networks to generate Bangla Handwritten digit generation. Because of using vanilla RNN there was no control over generating class specific characters. Antoniou et al [7] used cGAN to use it as a Data Augmentation method to increase the accuracy rate of Omniglot characters. Chang et al[8] used cycleGAN, another variation of vanilla GAN to generate chienese characters.

B. Artificial Neural Network

Artificial Neural Networks(ANNs) are a kind of artificial intelligence technique which mimics the human brain operation. ANNs consist of interconnected computer processors. Neural networks are programmed to learn by looking for relationships to create mathematical models and correcting these models automatically to refine them. Artificial Neurons are usually refereed as 'perceptron'. A perceptron will have many inputs and the inputs are distinctly weighted. The load can amplify or deamplify the substantive input signal depending on the context.

C. Convolutional Neural Networks

Convolutional Neural Networks (CNN) are biologically-instinct variants of ANNs. The connectivity of the neurons are mainly inspired by the visual cortex of animals.CNNs gained its popularity because of its use in the field of image classification. Also it played an significant role in the task of automatic caption generation for images. The structure of basic a basic CNN follows as: Convolution - Pooling - Convolution - Pooling - Fully Connected Layer - Output. Fig 2. shows the basic architecture of a convolutional neural network.

- Convolution takes the authentic data as input and creates feature map from it
- Pooling is the form of down-sampling the convolved image.
- Fully connected layers are the basic traditional ANNs.

D. Generative models and GANs

1) Generative Models: Generative models are a kind of unsupervised learning model which generates data using joint probability distribution.This types of model has the potential and power of learning features from image and datasets and then generating similar data.

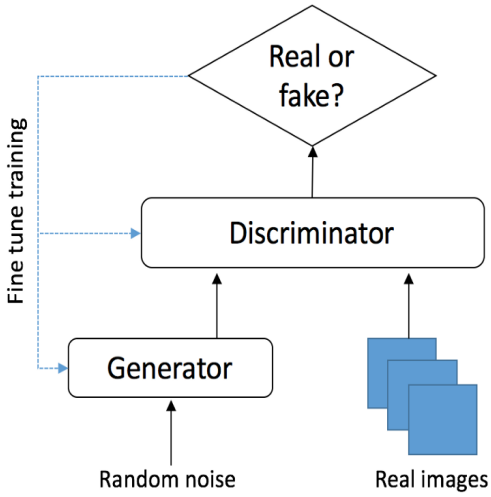


Fig. 2: Architecture of Generative Adversarial Networks

2) Discriminative Models: Discriminative models are used for modeling the reliance of independent variable on the dependent variable of a model using the conditional probabilities between them.

3) Generative Adversarial Networks: As already mentioned Generative Adversarial Networks are one of the most hot topic in recent times. GANs has the power to generate unique and robust images from a probabilistic distribution. Generative adversarial networks consists of 2 machine learning models

- Generative Model
- Discriminative Model

Neural network is used for the generative and discriminative model. Because of using neural nets for generative models, it requires smaller number of parameters than it usually requires for a neural net. Thus it does a very efficient job in generating data. Usually using unsupervised pre training on small datasets is effective. By unsupervised pre training we are meaning that training the model in an unsupervised manner. Usually a neural network requires a huge amount of data to be trained adequately in order to avoid overfitting which is a very crucial problem for neural networks. Fig 3 shows the exact mechanism behind GAN.

The first network of GAN is $G(Z)$ which is the generator and $D(Z)$ is second model which is the discriminator. The loss function of GAN is defined as

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

When the model reaches at its equilibrium point, which is the optimal point of the game minimax. The generator tries to generate data corresponding as the input data and the

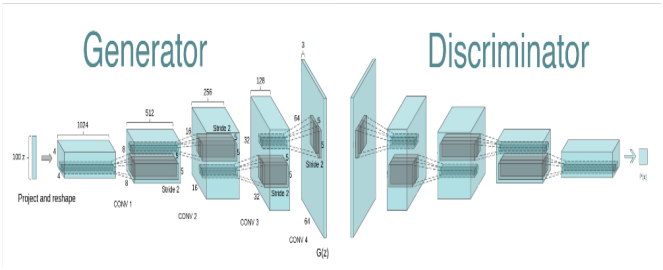


Fig. 3: Architecture of Deep Convolutional Generative Adversarial Networks

discriminator tries to predict whether image is original or not. When the discriminator reaches at equilibrium point it outputs the probability of its prediction 0.5 which clearly means that the discriminator was unable to predict the data.

4) Deep Convolutional Generative Adversarial Networks: In 2016, A variation of vanilla GAN was proposed namely Deep Convolutional Generative Adversarial Networks(DCGAN).[9] It is nothing but a stabilize GAN with some architectural variation and constraints. The key variation of DCGAN and vanilla GAN is that in vanilla GAN, authors used a multilayer perceptron neural network for generator and discriminator model, but in DCGAN, authors used a deep convolutional neural network for the generator and discriminator. The results were really promising. As it is known that CNN's are very good at feature extraction. The architecture for stable deep convolutional GANs has the following guidelines.

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Using batchnormalization for both generator and discriminator.
- There should not be any fully connected hidden layer for deep architectures.
- The last layer of generators should use tanh, other then the last layer all other layers should use ReLU activation function.
- For discriminator LeakyReLu activation function should be used.

Fig. 4 shows the architecture of DCGAN.

5) Conditional Generative Adversarial Networks: Conditional generative adversarial network (cGAN) is an expansion of the GAN. This work was first proposed by Mirza et al[10]. The problem with vanilla GAN's are that there are no control over which types data should the model generate. The model arbitarilly generates data based on the training datas. In cGAN, the authors have proposed an approach to condition the model to generate class specific datas. They conditioned both the generator and discriminator on given some extra information y. Here y is an additional information which can be class labels or data from different moralities. This y is fed into both generator and discriminator both as an auxilary layer. In generator prior input noise is $p_z(Z)$ and y are merged together as an hidden representation. The objective function of cGAN is stated as below

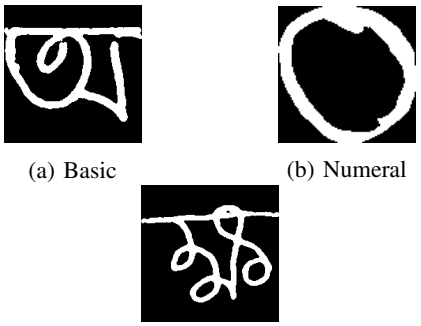
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$

In the original paper of cGAN they have used fully connected neural network as the architecture of the generator and discriminator. In our work we have created a variation of cGAN by mergind the architecture of DCGAN and cGAN. So we have used Deep Convolutional Generative Adversarial Networks for generating the characters.

TABLE I: Number of images in different datasets

Type	CMATERDB[1] [13]	ISI Dataset[2] [12]	BanglaLekha-Isolated
Basic Character	15,103	30,966	98,950
Numerals	6,000	23,299	19,748
Compound Characters	None	None	47,407

[1] CMATERDB dataset has 3 different datasets for basic characters,numerals and compound characters.
[2] ISI dataset has two different dataset for basic characters and numerals.



(a) Basic (b) Numeral (c) Compound

Fig. 4: Sample Images of BanglaLekha-Isolated

III. Dataset

The dataset we used in this paper is BanglaLekha-Isolated[11], Indian Statistical Institue[12], CMATERDB[13]. BanglaLekha-Isolated dataset contains sample of 84 different Bangla handwritten characters. From Indian Statistical Institue dataset and CMATERDB we have used the numerals and basic character. Fig. 5 shows some sample images that were used to train the model. Table I shows the in detail statistics about the dataset that was used in this work.

IV. Architecture of The Model

We have used the architecture that was used in[14], for image to image translation.

A. Generator

We used a Conv-BatchNorm-ReLu like encoder-decoder architecture for generator. We have used total 6 Conv-BatchNorm-ReLu architecture for encoder. In the encoder we have used 64,128,256,512,512,512 convoltuional layers. For

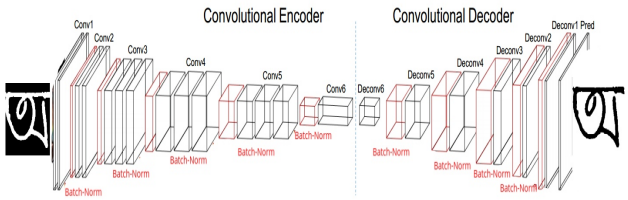


Fig. 5: Architecture of Deep Convolutional Generative Adversarial Networks

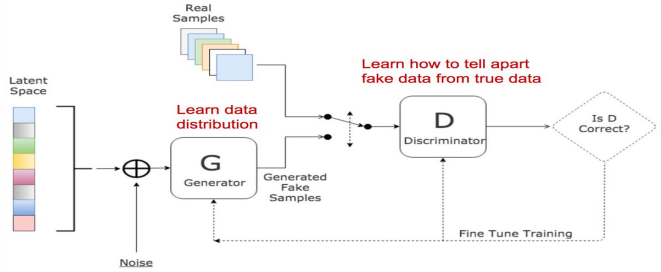


Fig. 6: Architecture of Conditional Generative Adversarial Networks

decoder we used the exact opposite architecture of the encoder. In the decoder 6 Conv-BatchNorm-ReLu architecture was used and there were 512,512,512,256,128,64 convolutional layers. For all the convolutional layers we have used ReLu activation function[15]. Architecture of the generator is given in Figure 5. The optimizer we used for this generator model was Stochastic Gradient Descent optimizer.

B. Discriminator

The first layer of the discriminator is a convolutional layer which has 64 filters, 5×5 stride value. The activation function used for discriminator is LeakyReLU, which was guided by the main authors of DCGAN. This followed by a Batch Nomralization and max pooling layer. Next, 128 filtered convolutional layer and the same activation function, batch normalization layer and max pooling layer was used. Same architecture was used for two more times with 512 and 1 Convolutional layers. We have used the similar optimizer for discriminator that was used in generator.

Architecture of the total conditional generative adversarial network is shown in Figure 6

V. Results and Analysis

We have run the described model for 1500 epochs and Fig. 7 shows us the loss value of generator and discriminator. After 1500 epochs the result generated from the generator seemed very much realistic. Figure 8 shows an example of a Bangla Basic characte Character. Figure 9 shows an example of generated Bangla Numeral. Figure 10 illustrates the generated samples of compound characters.

Fig. 7: Loss Graph for Generator and Discriminator

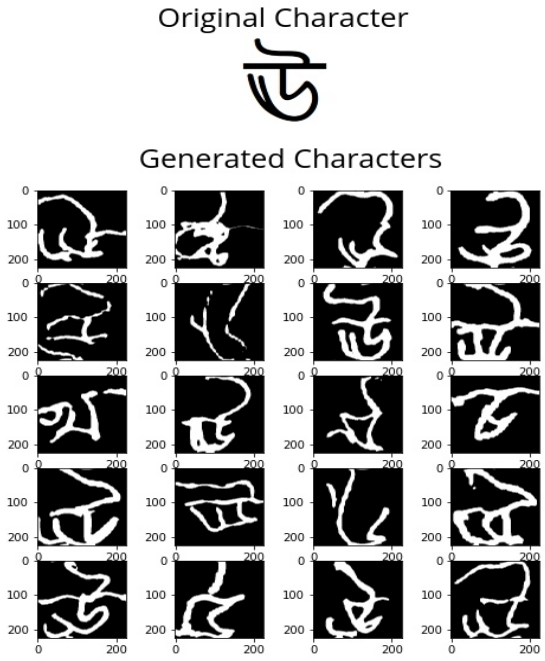


Fig. 8: Generated results for Bangla Basic Character

VI. Conclusions

In the field of artificial intelligence and machine learning dataset is the most crucial component. But gathering data's are one of the most hard work. But conditional generative adversarial networks can generate any kind of data of a specific class which can be used in machine learning. Our work can enhance the accuracy level of Bangla Optical Character recognition, which is one of the most difficult task. Our future works includes generating words using generative adversarial

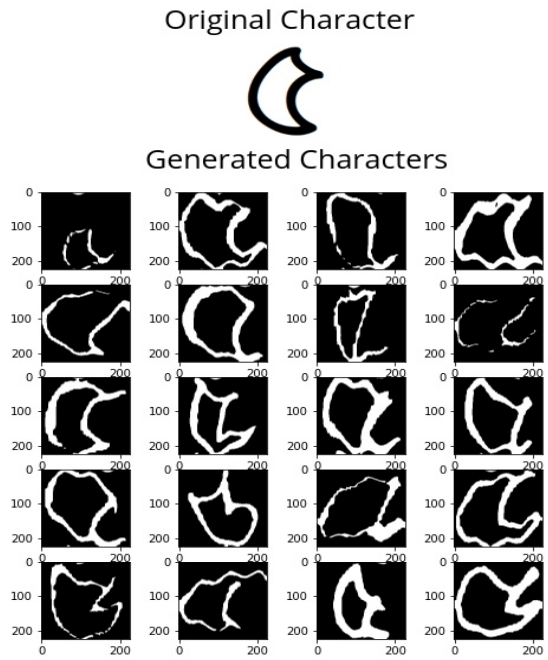


Fig. 9: Generated results for Bangla Numeral Character

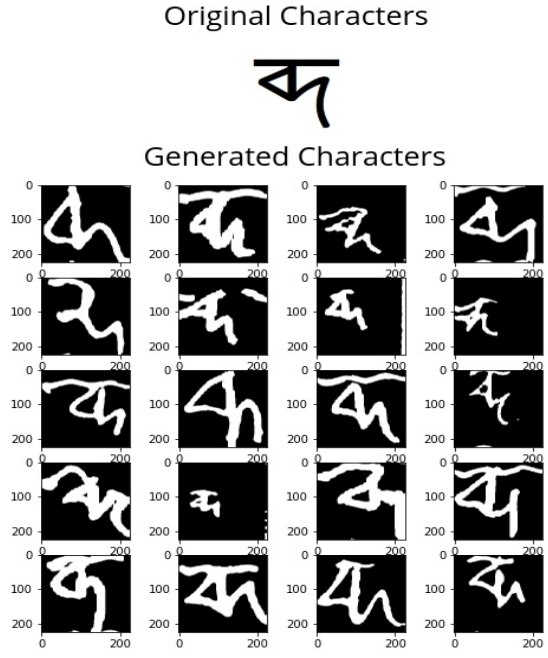


Fig. 10: Generated results for Bangla Compound Character

networks.

REFERENCES

[1] N. Sünderhauf, F. Dayoub, S. Shirazi, B. Upcroft, and M. Milford, "On the performance of convnet features for place recognition," *arXiv preprint arXiv:1501.04158*, 2015.

[2] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," 2007.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[4] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[6] M. F. Sikder, "Bangla handwritten digit recognition and generation," in *Proceedings of International Joint Conference on Computational Intelligence*. Springer, 2020, pp. 547–556.

[7] A. Antoniou, A. Storkey, and H. Edwards, "Data augmentation generative adversarial networks," *arXiv preprint arXiv:1711.04340*, 2017.

[8] B. Chang, Q. Zhang, S. Pan, and L. Meng, "Generating handwritten chinese characters using cyclegan," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018, pp. 199–207.

[9] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[10] M. Mirza and S. Osindero, "Conditional generative adversarial networks," *Manuscript: https://arxiv. org/abs/1709.02023*, vol. 9, p. 24, 2014.

[11] M. Biswas, R. Islam, G. K. Shom, M. Shopon, N. Mohammed, S. Momen, and M. A. Abedin, "Banglalekha-isolated: A comprehensive bangla handwritten character dataset," *arXiv preprint arXiv:1703.10661*, 2017.

[12] U. Bhattacharya and B. B. Chaudhuri, "Handwritten numeral databases of indian scripts and multistage recognition of mixed numerals," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 3, pp. 444–457, 2009.

[13] "Cmaterdb - CMATERdb: The pattern recognition database repository," http://www.findbestopensource.com/product/cmaterdb, accessed: 2017-02-20.

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[15] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

Assessment of Bangla Descriptive Answer Script Digitally

Md Gulzar Hussain*, Sumaiya Kabir†, Tamim Al Mahmud‡, Ayesha Khatun§, Md Jahidul Islam¶
Department of Computer Science and Engineering*
Green University of Bangladesh, Dhaka-1207, Bangladesh
gulzar.ace@gmail.com*, sumaiya@cse.green.edu.bd†, tamim@cse.green.edu.bd‡,
ayesha@cse.green.edu.bd§, jahidul.jnucse@gmail.com¶

***Abstract*—Answer script evaluation is an essential part of the student evaluation process in the education system. In an exam, students need to answer subjective and objective questions. In educational institutes, instructors need to evaluate the answer script manually to evaluate the students. In Bangladesh, the number of students and institutes are increasing day by day. For this reason, it is becoming hard to evaluate the answer script in a perfect way by the instructors. So it is necessary to find a way to evaluate the answer script automatically. Many techniques are proposed for the English language, but we didn't find any for Bangla language. Our paper proposed a way to evaluate Bangla subjective answer scrips automatically by keyword matching and linguistic analysis. Using our proposed model we have tested on answer scripts of 20 questions and found the minimum relative error of 1.8%. 15 teachers and 10 students volunteered to evaluate the answer scripts.**

Keywords: Bangla Language, Bangla Subjective Script, Evaluation, Keyword Matching, Automatic Evaluation, Answer Script.

I. Introduction

To test the skills and knowledge of individuals, examination systems are designed. There are various kinds of tests or examination systems all over the world, such as multiple choice questions, subjective, etc. Objective evaluation is a way of questioning that has one correct answer. In the other hand, subjective evaluation can have more than one correct answer. Like other countries, Bangladesh also follows both of these evaluations. In Bangladesh there are around 23,907,151 students overall in primary, secondary, post secondary level of education in 2015 and primary language used in these education levels are Bangla and English [1]. But there are not enough teachers or instructors for those students. It is becoming hard to evaluate these students for these teachers or instructors. In 2015, the teacher & student ratio was 1:41 [2] [3]. Teaching students, setting questions, evaluating the answer script of the students is difficult for the teachers due to this ratio. If some of these tasks can be performed then it will make life easier for the instructors. Not only in the education system but also in the various job entrance examination is done in a subjective way. These answer scripts also need to be evaluated. So if we can asses the answer script automatically then the evaluation system will be more smooth for Bangladesh.

The rest of the paper is organized as follows: **Section II** discusses previous works. Methodology discussed in **Section III**. Result analysis and discussion demonstrated in **Section IV**. And finally **Section V** refers the conclusion.

978-1-7281-5242-4/19

II. Previous Works

Generally, answer scripts evaluation is a difficult task. It becomes harder for Bangla language. Various works are found for English but none is found for Bangla Text. Here we will discuss some previous works to evaluate subjective answer script.

Authors of [4] suggest alternative sentence generator method to produce an alternative model response by linking the technique to a synonymous dictionary. In the matching stage, they proposed combination of three algorithms, Commons Words (COW), Longest Common Sub-sequence (LCS) and Semantic Distance (SD), which were used effectively in many Natural Language Processing systems and produced effective outcomes. Hyperspace Analog to Language (HAL) procedure and Self-Organizing Map (SOM) method is used to evaluate students answer script in paper[5]. Kohonen Self-Organizing Maps clustering technique is applied to the vector in their suggested system. Authors of [6] observed that the semantic Enhanced NLP based technique outperforms easy lexical matching techniques. This application mechanism offers an automatic answer assessment based on the keyword given by the moderator to the application in the form of the input that will provide equal mark distribution.

In paper [7], authors proposed a Natural Language Processing (NLP) based method for evaluating of the answer script. They used a keyword based summarizing technique to generate a summary of the answer script. Authors of [8] proposed a syntactical-relation based feature extraction technique to evaluate descriptive type answer scripts. Their method contains steps like question-classification, answer-classification, and answer-evaluation of subjective answers of students and grade with a suitable score. Advanced machine learning techniques and methodologies based on a new model is proposed by Prakruthi et. al. in [9]. They did it for Optical Character Recognition based work involving supervised learning technique. An implementation that uses machine learning to evaluate the answers scripts is proposed in paper [10].

Authors of [11] proposed evaluation procedure in a semi-automated manner where subjective questions are supple-

mented with model answer points. Their suggested framework also includes provisions of reward systems and penalties.

III. Methodology

Working methodology for our proposed system is discussed in this section. Our system follows the given system diagram in Fig 1 to evaluate the Bangla answer script of the examinee.

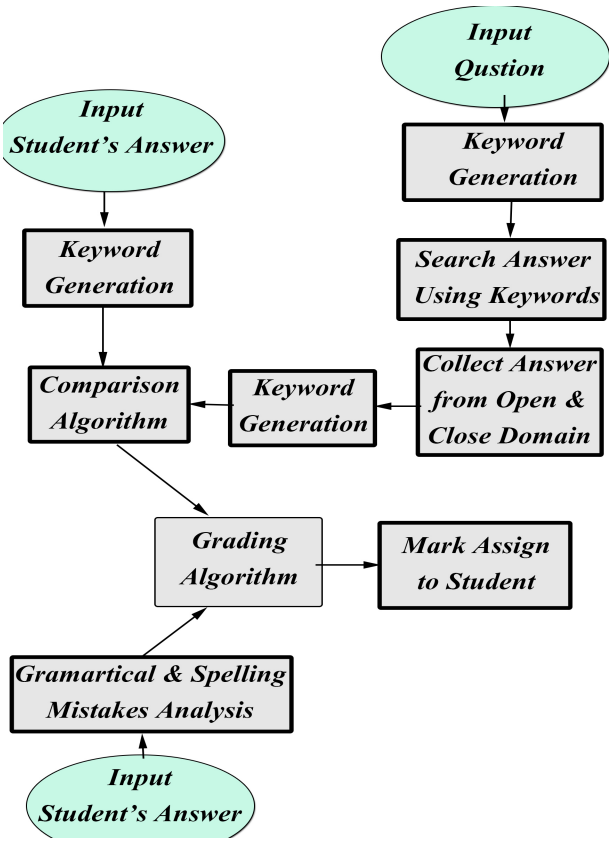


Fig. 1. System Workflow of proposed system

In the system flow diagram, we can see that step "Keyword Generation" repeated for the analysis of question, students answer script, and answer collected from open & close domain. This is an important step in our proposed system. We can also see that to evaluate the student's answer, we searched keywords generated from question in the open domain and closed domains. Here open domains include World Wide Web, various web pages, blogs, Wikipedia etc. and closed domain consist of a specific category or specific answer to the question. A comparison algorithm is proposed for comparing our generated answer keywords and keywords generated from the student's answer. These various steps are elaborated below:

A. Keyword Generation

Our proposed system simply takes an answer from the students as a text or document file. These answers are written by students in a document file. We are not taking handwritten answer scripts in consideration due to reducing the complexity. In this step, we process the text very carefully. Sub steps of Fig 2 are taken in this step.

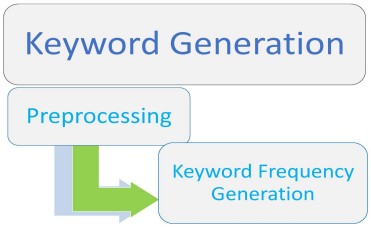


Fig. 2. Sub-steps in the Keyword Generation

1) Preprocessing: As answers collected from various domains and students contain many unnecessary words, stop symbols, punctuation symbols, emotions symbols etc. These are noise for our unprocessed data. We remove these special characters like #, &, % etc. emojis like :), :P etc, stop words etc. We also remove the articles, punctuations etc. to simplify the text. After preprocessing keywords are generated and we use the following sub-steps to generate their frequency.

2) Keyword Frequency Generation: After preprocessing all the text data we need to identify every keyword and count their frequency ratio. To do that we used algorithm 1.

Algorithm 1: Algorithm to Generate Keyword Frequency

```
Answer = Answer Script after preprocessing;
Frequency_Array = NULL;
Number_Keywords = Number of keywords in the
  Answer;
for Words in Answer do
    if Word found in Frequency_Array then
        Frequency_Array[word] =
          Frequency_Array[word] + 1/Number_Keywords;
    else
        In the Frequency_Array add that word as index ;
        Frequency_Array[word] = 1/Number_Keywords;
    end
end
```

Frequency Array is formed after applying algorithm 1 where frequency of every word is calculated. There will be no stop words, punctuation, irrelevant words in this array as all of them will be removed after preprocessing.

B. Searching & Collecting Answers Using Keywords

To assessment, student answer system will need a standard answer for a question. Answers can be from an open or close domain. An instructor can provide the answer manually or system can collect the answer from the internet or other resources. Our system will parse data from online knowledge-based websites like Wikipedia etc. The MediaWiki action API is a good example of web service that provides access to certain wiki features such as authentication, page operations, and search where the entry point is bn.wikipedia.org/w/api.php or many others. Our proposed system will collect data depending on the keywords from the World Wide Web or any other local resources. As an example, if the question is Bangladesh

sampark likhuna (Write about Bangladesh.) then it will search about Bangladesh and collect answers from these domains.

C. Comparison Algorithm to compare Two Results

When keywords and their frequency ratio is calculated for a student's answer script and searched answer, then we need to compare them for scoring the student's answer. To do that algorithm 2 is proposed.

Algorithm 2: Comparison Algorithm

KeywordScore = 0;
SAFrequency = Frequency Result Of Student's Answer;
SRFrequency = Frequency Result Of Searched Answer;
LengthSR = Size of SRFrequency;
WeightSA = Empty Array;
Sort SRFrequency in decending order;
for *word in SAFrequency* **do**
  if *word is in SRFrequency* **then**
    'word' added as index in WeightSA;
    if *SAFrequency[word] > SRFrequency[word]* **then**
      $WeightSA[word] = 1$;
    else
      $WeightSA[word] = \frac{SAFrewuency[word]}{SRFrequency[word]}$;
    end
    $SAFrewuency[word] = 0$;
    $SRFrequency[word] = 0$;
  else
  end
end
NeededWord = 0;
for *word in SRFrequency* **do**
  if *SRFrequency[word]* $! = 0$ **then**
    $WeightSA[word] = -1 * SRFrequency[word]$;
    $SRFrequency[word] = 0$;
    $NeededWord+ = 1$;
  else
  end
end
UnnecessaryWord = 0;
for *word in SAFrequency* **do**
  if *SAFrequency[word]* $! = 0$ **then**
    $WeightSA[word] = -1 * SAFrequency[word]$;
    $SAFrequency[word] = 0$;
    $UnnecessaryWord+ = 1$;
  else
  end
end

D. Grammatical & Spelling Mistake Analysis

Grammatical and Spelling mistake is very important for evaluating any kind of answer script. It is also important because students can write just the keywords required for a correct answer. If grammatical mistakes are checked then it will come to the knowledge of the system that the student is tried to cheat. We can see in the evaluation process grammar and spelling checking is done in many proposed method. But as we are working with Bangla language in which case such resources are limited and we are using Akkhor Bangla Spell and Grammar Checker [12] in our proposed method, which one is a tool to check spelling and grammatical mistakes. Hence we are proposing a simple keyword-frequency-based process with the use of this tool. This tool will return a score depending on the grammatical and spelling mistakes. That score will help the evaluation system to evaluate the student's answer script. The score is calculated based on algorithm 3.

Algorithm 3: Algorithm for Scoring Grammatical and Spelling Mistakes

NumberOfWord = Number of total words in Student Answer Script;
NumberOfSentence = Number of total words in Student Answer Script;
SMistakes = 0;
GMistakes = 0;
TGSMScore = Total Grammatical & Spelling Mistake Score which is initially 0;
for *Words in Student Answer Script* **do**
  if *word is in Student Answer Frequency* **then**
    $SMistakes = \frac{WeightSA[word]}{NumberOfWord} + SMistakes$;
  else
    $SMistakes = \frac{1}{NumberOfWord} + SMistakes$;
  end
end
for *Sentences in Student Answer Script* **do**
  $GMistakes = \frac{1}{NumberOfSentence} + GMistakes$;
end
$TGSMScore = \frac{SMistakes+GMistakes}{SMistakes*GMistakes}$

E. Mark Assigning to Students

After performing algorithm 2 and 3, we will get some values of parameters LengthSR, NeededWord, Un-necessaryWord, WeightSA, and TGSMScore. We need to assign marks to the students depending on the answer script. These values will be used in the grading algorithm step to assign mark to the student depending on his or her answer script. In the grading algorithm we are considering that out of the total mark 80% will be allocated to the answer and 20% will be for the grammatical and spelling mistakes. Exam authority will be able to change the value as their need.

In this step, calculation of the final mark of a student is done using the weights which are set in the comparison algorithm and score obtain from algorithm 3. Mark assignment is done by following grading algorithm 4. This algorithm takes consideration the score obtain from total grammatical and spelling mistakes and the number of necessary and unnecessary words to assign the mark to the answer.

Algorithm 4: Grading Algorithm to Assign Mark

StudentMark $= 0$;
FullMark $=$ N;
ActualMark $= 0.8 * N$;
GSMark $= 0.2 * N$;
for *Words in WeightSA* **do**
  if *WeightSA[word] is positive* **then**
    $ActualMark = (N * WeightSA[word]) + ActualMark$;
  else
    $ActualMark = \frac{WeightSA[word]}{NeededWord+UnnecessaryWrod} + ActualMark$;
  end
end
$GSMark = TGSMScore * GSMark$;
FullMark $=$ ActualMark + GSMark;

TABLE I
ABSOLUTE AND RELATIVE ERRORS FOR PROPOSED SYSTEM

Number of Teachers	Average Score given by Teachers	Score given by the system	Absolute Error	Relative Error
5	8.56	8.1	0.46	5.37%
10	8.25	8.4	0.15	1.81%
15	8.4	8.7	0.3	3.57%

IV. Result Analysis

For experimenting our system, we gave 20 questions for answering to 10 students. Every students answered two questions and each answer contain 300-350 Bangla words. Each of the answer scripts are evaluated by our volunteered teachers and assigned a score to each of the answer. Each of the questions was of 10 marks. We also evaluate the answer scripts using our proposed system. Based on the scores, we have calculated the Absolute and Relative Errors, and found table I.

We calculated the Absolute error using the formula 1 and relative error using formula 2.

$AbsoluteError = ActualValue - MeasuredValue$ ...(1)
$RelativeError = \frac{AbsoluteError}{ActualValue} * 100$ .....(2)

Where in formula 1 actual value is the score given by the teachers and measured value is the score given by the system. We also find the relative error of answer script of each or the questions are given in fig 3 which is the graph of Questions Vs Relative error.

As resources in Bangla Language are hard to find and there is no work found in past our work is just an initiative. But the relative error which is observed in our system can be accepted as it is just the beginning. We hope to improve our work in the future. In the future implementation of the whole idea is needed to make it automated and faster. New parameters can be added to the system to make it more reliable and synonyms of words can be checked. Machine Learning algorithms can be used to make the system more efficient and effective. Our proposed method should be implemented for various type of sentences. Input should be taken from handwritten answer scripts to make it more applicable in the real word.

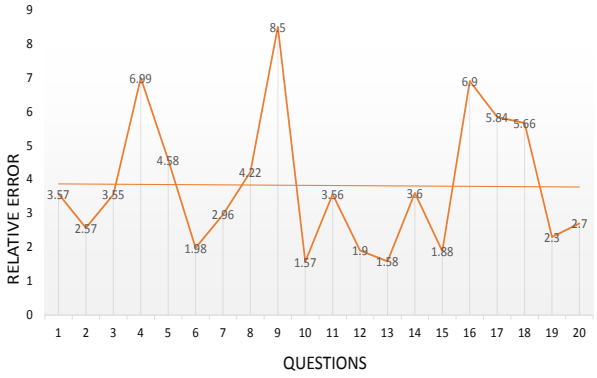


Fig. 3. Questions Vs Relative error

V. Conclusion

We proposed a basic method to evaluate Bangla subjective answer script for the first time. The relative error is under 10% of our proposed methodology which is quite satisfactory. As it is just the beginning, there are still many options or ways to improve the experimental methodology we proposed. We know that working for the Bangla language is more difficult then English language due to its minimum amount of resources. We hope our proposed system will save some times and utilize the current resources.

References

[1] Wikipedia. (2015) Education in bangladesh. [Online]. Available: https://en.wikipedia.org/wiki/Education_in_Bangladesh
[2] BANBEIS. (2017) Bangladesh education statistics 2016. [Online]. Available: http://lib.banbeis.gov.bd/
[3] AsiaNewsNetwork. (2017) Teacher-student ratio worsens in bangladesh. [Online]. Available: http://annx.asianews.network/content/teacher-student-ratio-worsens-bangladesh-46376
[4] A. Benomran and M. Ab Aziz, "Automatic essay grading system for short answers in english language," *Journal of Computer Science*, vol. 9, pp. 1369–1382, 09 2013.
[5] K. Meena and L. Raj, "Evaluation of the descriptive type answers using hyperspace analog to language and self-organizing map," in *2014 IEEE International Conference on Computational Intelligence and Computing Research*, Dec 2014, pp. 1–5.
[6] K. Wangchuk, "Automatic answer evaluation: Nlp approach," 05 2016.
[7] M. Rahman and F. Hasan Siddiqui, "Nlp-based automatic answer script evaluation," vol. 4, pp. 35–42, 12 2018.
[8] V. Nandini and P. Uma Maheswari, "Automatic assessment of descriptive answers in online examination system using semantic relational features," *The Journal of Supercomputing*, 05 2018.
[9] M. Prakruthi S T, "Automated students answer scripts evaluation system using advanced machine learning techniques," *International Journal for Research in Applied Science and Engineering Technology*, vol. 6, pp. 1794–1797, 06 2018.
[10] P. Sinha and A. Kaul, "Answer evaluation using machine learning," 03 2018.
[11] C. Roy and C. Chaudhuri, "Case based modeling of answer points to expedite semi-automated evaluation of subjective papers," in *2018 IEEE 8th International Advance Computing Conference (IACC)*, Dec 2018, pp. 85–90.
[12] Akkhor. (2019) Akkhor bangla spell and grammar checker. [Online]. Available: http://www.akkhorbangla.com/

Real-time Bangladeshi Currency Detection System for Visually Impaired Person

Md. Ferdousur Rahman Sarker[1], Md. Israfil Mahmud Raju[2], Ahmed Al Marouf[3], Rubaiya Hafiz[4], Syed Akhter Hossain[5]
Department of Computer Science and Engineering
Daffodil International University
Dhaka, Bangladessh
Email: {rahman15-6783, mahmud15-7141, marouf.cse, rubaiya.cse}@diu.edu.bd
aktarhossain@daffodilvarsity.edu.bd[5]

Munim Hossain Khandker Protik
RDRS Rehabilitation Centre for Visually Impaired Children,
Lalmonirhat, Bangladesh
Email:protik13@gmail.com

Abstract— **This paper presents a real-time Bangladeshi currency detection system for visually impaired persons. The proposed system exploits the image processing algorithms to facilitate the visually impaired people to prosperously recognize banknotes. The recent banknotes of Bangladesh have blind embossing or blind dots, which could be effective to recognize the value of the bill by touching. As the embossing fades away in the long-term used notes, detecting right value of the banknote using image processing algorithms could be considered as a challenging task. Particularly in Bangladesh, each banknote seems similar using the direct exertion of simplified image processing algorithms. In this paper, a recognition system was implemented that can detect Bangladeshi banknote in different viewpoints and scales. The detection system is also able to detect currency those are rumpled, decrepit or even worn. The detection system includes image preprocessing, image analysis and image recognition. To enhance the determination of currency recognition, the descriptor of an individual input scene is matched with various training images of the same category. After that, by analyzing their matching result it recognizes the currency with higher confidence. For real-time recognition, we have deployed the system into a mobile application.**

Keywords—Bangladeshi Currency Recognition, Feature matching, Image Processing, Image Analysis.

I. INTRODUCTION

Globally, it is estimated that approximately 1.3 billion people live with some form of distance or near vision impairment. With regards to distance vision, 188.5 million have a mild vision impairment, 217 million have moderate to severe vision impairment, and 36 million people are blind according to the World Health Organization[1]. In Bangladesh, there are about 40,000 visually impaired. It is a matter of concern that 80% of the visually impaired persons live in rural areas where the treatment facilities are very poor as 90% of the doctors and paramedics are urban-based. With these statistics, it is to be mentioned that the increased number of visually impaired employees in the government and private job sectors is highly impressive.

One of the main problems suffered by a visually impaired person to identify paper currencies because of the similarity of paper size and texture between different banknotes. As currencies are the commonly used stuff in everyday life, understanding the value of the banknotes is a very important task for them. Therefore, we have proposed a real-time system that will help them to recognize the currencies and resolve this crisis to make visually impaired people feel confidence in the financial dealings, not depending on others.

There are two trends in currency recognition: scanner-based and camera-based. Scanner-based systems are quite hard to carry and the recognition rate is not up to the mark. While camera-based systems can be developed with the help of modern image processing techniques and each smartphone has camera features, which makes it available to the visually impaired persons. Therefore, the camera-based systems will be much easier to use and feasible for the target users. Developing a feasible camera-based mobile application which will open up the camera and recognize the banknotes in real-time.

Different image processing systems such as object recognition [2], shape recognition [3], action recognition, real-time hand gesture recognition [4, 5] are adopted in recent years. Researchers' had created a fairground to utilize different machine learning-based algorithms to solve similar banknote detection systems. Irrespective of the noisy images, the recognition depends on the key values extracted from the images. Template matching [6], shape descriptors [7] etc. are outperformed by regular feature descriptors such as scale-invariant feature transform (SIFT) [8], speeded-up robust feature (SURF) [9], maximally stable external regions (MSER) [10] etc. These feature descriptors are capable enough to solve the image rotation, scale or orientation. In Bangladesh, the available denominations are 1000 BDT, 500 BDT, 100 BDT, 50 BD, 20 BDT, 10 BDT, 5 BDT and 2 BDT. For helping the blind community to understand the value of the banknotes, the authority had placed blind embossing or blind dots on the banknotes. The number of blind dots differs in different banknotes. It has five dots in 1000 BDT, four dots in 500 BDT, three dots in 100 BDT, two dots in 50 BDT and one dot in 20 BDT notes. Though these blind dots are very helpful in case of new banknotes, but whenever the notes get dirtier or worn, it is very difficult to perceive the number of dots. Therefore, applying image processing algorithms to differentiate between the banknotes by keypoint extraction and describing them accordingly.

In this paper, camera-based Bangladeshi paper currency is trained to be recognized using simplest image processing utilities which makes the processing time very short with acceptable accuracy. We have applied the Oriented FAST and Rotated BRIEF (ORB) [11] based keypoint extraction and the system has the ability to treat papers captured partially and under different lighting conditions and perspectives. In the rest of the paper, section II presents the related works found in the literature related to currency recognition and mobile applications. In section III a detailed description of the implementation has described with a detail explanation of the

978-1-7281-5242-4/19

methods. Section IV presents the experimental results and comparison between different algorithms. Finally, the conclusions will be given in Section VI.

II. Related Work

In this section, we have discussed on the literature review on the problem statement of this paper. Considering a real-time system to detect the Bangladeshi currency accurately, we have divided this section into two parts: related works in currency detection using image processing and related works on real-time mobile application for the same problem.

Several methods have been adopted by the researchers for solving this problem. L. J. Feng et al. [12] proposed an algorithm for real-time detection of Chinese currencies. A modified Kohonen Network is developed for recognizing Chinese paper currencies for the experiments by them. The method applied could be used for practical currency sorting system. A hardware implementation is performed in [13] for paper currency number recognition utilizing the CIS scanning circuits and ARM.

An edge-based detection algorithm is proposed in [14] for paper currency by J. Ye. The method divides the currency image into several overlapping subzones and within each subzones, the defect feature is calculated to estimate the stage of contamination. The method proved to be robust while applied to low quality paper currency.

K. Fanhui et al. [15] proposed a Gaussian Mixture Model (GMM) based paper currency recognition. The applied method in [15] based upon the structural risk minimization (SRM) to develop a faster system. The experimentation shows that GMM exploiting the SRM provides more flexibility and leads to an improved result on Chinese paper currency recognition.

Y. Weiqi et al. [16] proposed a fast recognition system for paper currency numbers. The method captures the 24-bits of color images using CCD camera, then outputs number clusters through the process of segmentation by gray ridge-vale algorithm. The orientation by projection and character recognition by the structure-analyzing algorithm is implemented as data processing. The experimental analysis shows satisfactory results as the system recognize the paper currency with high rate and fast recognition speed.

An image processing based fake banknotes detection system is proposed in [17] which only works on denomination of 1000 BDT. They got 63.34% accuracy for the experimental analysis presented using edge detection methods. Focusing on the fake or counterfeit Bangladeshi currency in [18], presented by Z. Ahmed et al., and presented a feasibility analysis. Similar Bangladeshi currency detection system proposed in [19] by M. M. Rahman, utilizing similar approach, but with 89.40% accuracy on white paper background and 78.40% accuracy tested on complex background. Similar machine vision based approaches [20, 21], hardware implementation using PIC or ATMega88 devices [22] are presented. Among the stated works, [19] has recorded the highest accuracy, but with a proper white background only.

The above mentioned algorithms and methods were presented for Chinese or Bangladeshi paper currency and developed system using machine learning. The idea of developing these sort of systems of visually impaired persons is missing in the literature. Therefore, the need of a real-time system for visually impaired persons is quite high and in this paper, we try to mitigate the gap in the literature.

In this paper, we have exploited the ORB method for keypoint detection and feature descriptions. The method experimentally shows better accuracy in recognition with faster processing time.

III. Implementation

The following sections present the main processing stages of the implemented Bangladeshi Currency recognition system (overview shown in Fig. 1). To speed up the development, the Open Source Computer Vision (OpenCV) library was used.

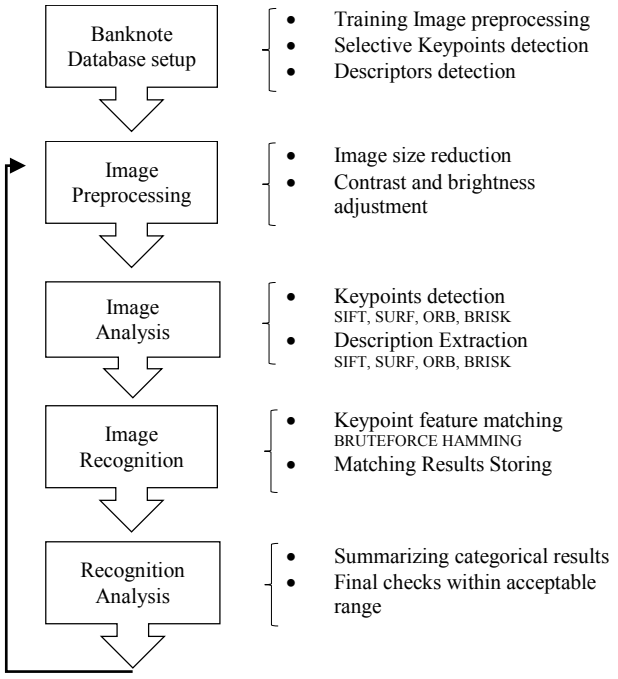


Fig. 1. Overview of the main processing stages

A. Reference Image Database Setup

In order to recognize the banknotes, a database of valid instances must be computed. This database contains the descriptors associated with the keypoints generated by OpenCV. To improve the detection we used Five training images of each banknotes. In each folder, there are different images of a currency to get matching results for different variations.

Fig. 2. Training images stored by category of different notes.

B. Image Preprocessing

In Bangladeshi currency, one side of each note are quite similar. Processing that tough side has the chance to get

inaccurate results. Therefore, the first step is to exclude that side if found. It will ask to place another side of the note. The whole thing start after the conversion of the input image to Grayscale. By gray-scaling the image will reduce inappropriate results due to lighting variations. System will automatically explore all the training folders and for each folder it will detect the keypoints of the training image using ORB Keypoint Detection and also extract the descriptors using ORB Descriptor Extractor. Descriptors of training images will be alive until the system is terminated. This descriptors will be used later to match with the descriptor of input images.

ORB is based on BRIEF. However, ORB addresses the detection, the orientation assignment and the descriptor extraction phases rather than only the descriptor (i.e. BRIEF). ORB adopts a multi-scale approach

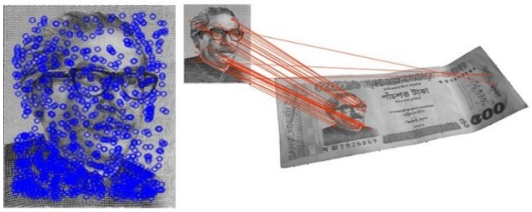


Fig. 3. Impact of preprocessing stage and detection of tough side of banknote

C. Image Analysis

After preprocessing, we detect the keypoints of input image for future recognition using ORB Keypoint Detection. As the system runs in real-time detection so ORB is the fastest. Also the Descriptor Extractor extract its description using ORB.

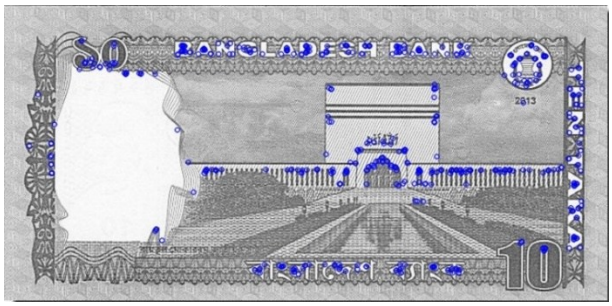


Fig. 4. Keypoints detection on an input image for future feature matching

Figure 4 shows the point detected for matching by ORB Detection. Though ORB analysis on small amount of keypoints, but for our currency recognition system the detected keypoints are enough to start matching as there are five images for each banknote. Detected descriptor of input image will be stored and the system will start matching in Image Recognition part.

D. Image Recognition

After input image analysis the system will now match the descriptors of training images that we generated from Preprocessing with the descriptor of input image. The descriptor of input image will be matched one by one with the training descriptors and calculate the matching results. For matching the descriptors we will be using OpenCV Descriptor Matcher. Between different types of descriptor method, we will be using Brute force Hamming algorithm for the matching phase. Each matching results will be stored for the analysis part, where the system will analysis between the results to return the most accurate matching banknote.

Fig. 5. Descriptor matching between input image descriptor and training image descriptor

E. Recognition Analysis

Here we find the maximum results between the additions of each set of banknotes. First we calculate the sum of matching points of all training images of each category of banknotes. If a, b, c, ..., h are different training folders of 2 Taka, 5 Taka, 10 Taka, ..., 1000 Taka banknotes respectively containing 5 training images each then x_a, x_b, x_c, ..., x_h are the summations of matching points of their set of training images.

$$\sum_{a=1}^{5} x_a = a_1 + a_2 + a_3 + a_4 + a_5$$

$$\sum_{b=1}^{5} x_b = b_1 + b_2 + b_3 + b_4 + b_5$$

$$\text{----------}$$

$$\text{----------}$$

$$\sum_{h=1}^{5} x_h = h_1 + h_2 + h_3 + h_4 + h_5$$

Finally, the system will measure the banknote category containing the maximum matching points.

$$\text{Result} = \max(x_a, x_b, x_c, x_d, x_e, x_f, x_g, x_h)$$

As the system will measure the maximum matching set, chances of inaccuracy will reduce dramatically.

IV. Experimental Results

The system contains total 40 training images and was tested in two different perspective views, Ideal and Distortion.

Fig. 6. Different perspective views (Ideal perspective in Left and Distortion perspective in Right)

While testing, we used 87 images of Ideal perspective and 97 images of Distortion perspective. Table I and Table II illustrates the True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) values for ideal and distortion perspective, respectively, applying the ORB detector, descriptors and brute force hamming matcher.

The training time (1.12s) and average matching time for the ideal case is very minimal as it shows 100% accuracy. Table III shows the accuracy and Average matching time for other methods, which provides a clear perception that our implemented ORB based methods are significantly faster (0.17s). The same experimentation is performed over the distortion perspective using noisy images having training time (1.13s). Hence, in this case, the matching time is also significantly faster (0.19s) than the other methods.

TABLE I
CONFUSION MATRIX IN IDEAL PERSPECTIVE, ORB DETECTOR, ORB DESCRIPTOR, BRUTEFORCE HAMMING MATCHER

N = 87	Predicted **YES**	Predicted **NO**
Actual **YES**	TP 53	FN 0
Actual **NO**	FP 0	TN 34

TABLE II
CONFUSION MATRIX IN DISTORTION PERSPECTIVE, ORB DETECTOR, ORB DESCRIPTOR, BRUTEFORCE HAMMING MATCHER

N = 97	Predicted **YES**	Predicted **NO**
Actual **YES**	TP 62	FN 1
Actual **NO**	FP 3	TN 31

TABLE III
COMPARISON WITH OTHER METHODS

Detector	**Descriptors**	**Matcher**	**ACC**	**Avg. Matching Time**
Ideal Perspective				
SURF	SURF	BRUTEFORCE	1	15.1s
SIFT	SIFT	BRUTEFORCE	1	10.9s
BRISK	BRISK	BRUTEFORCE	0.69	0.96s
ORB	ORB	BRUTEFORCE	1	**0.17s**
Distortion Perspective				
SURF	SURF	BRUTEFORCE	0.86	17.1s
SIFT	SIFT	BRUTEFORCE	0.86	12.4s
BRISK	BRISK	BRUTEFORCE	0.45	1.15s
ORB	ORB	BRUTEFORCE	0.96	**0.19s**

The real-time implementation of the system is demonstrated in [23] while using the mobile application practically. The exploited method has been implemented using OpenCV package of Android. The mobile application need to use the camera features available in the device.

V. CONCLUSION

In this paper, we try to formalize a real-time Bangladeshi currency detection system implemented over mobile application. We have applied the widely used ORB based feature descriptor for recognizing the traditional Bangladeshi paper currencies. The average recognition rate for each of the different types of banknotes are documented in the experimental results. The recognition rate are higher than any other methods applied for experimentation and the average matching rate is also quite satisfactory considering a real-time system. The presented system could be very helpful for the visually impaired personals, who can use the mobile application to recognize the banknotes very accurately.

REFERENCES

[1] R. R. A. Bourne, S. R. Flaxman, T. Braithwaite, M. V. Cicinelli, A. Das, J. B. Jonas, "Vision Loss Expert Group. Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: a systematic review and meta-analysis", Lancet Glob Health. 2017 Sep;5(9):e888–97.

[2] Z. Q. Zhao, P. Zheng, S. Xu, X. Wu, "Object detection with deep learning: A review", CoRR, abs/1807.05511, 2018.

[3] D. Zhang, G. Lu, Review of shape representation and description techniques, Pattern Recognition 37 (1) (2004) 1–19.

[4] A. A. Marouf, S. Shondipon, M. K. Hasan, H. Mahmud, "4Y model: a novel approach for finger identification using KINECT". In: IEEE 2nd International Conference on Recent Trends in Information System, pp. 183–188. IEEE, Kolkata, 2015.

[5] A. A. Marouf, M. F. R. Sarker, S. M. T. Siddiquee, "Recognizing Hand-based Actions based on Hip-Joint centered Features using KINECT", 2nd International Conference on Electrical & Electronic Engineering (ICEEE), 27-29 December, 2017.

[6] D. A. Zuehlke, T. A. Henderson, S. A. H. McMullen, "Machine learning using template matching applied to object tracking in video data", Proceedings Volume 11006, Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications; 110061S, 2019.

[7] M. Yang, K. Kpalma and J. Ronsin. "A survey of shape feature extraction techniques", Pattern Recognition, (2008), pp. 43-90.

[8] T. Lindeberg, "Scale invariant feature transform," Scholarpedia, vol. 7, no. 5, p. 10491, 2012.

[9] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In European Conference on Computer Vision, May 2006.

[10] M. Donoser, H. Bischof, "Efficient maximally stable extremal region (mser) tracking". IEEE International Conference on Computer Vision and Pattern Recognition, 553–560, 2006.

[11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: an efficient alternative to SIFT or SURF. In Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV), volume 13, 2011.

[12] L. Jia Feng, L. Song Bo, T. Xiang Long, "An Algorithm of Real-Time Paper Currency Recongnition", Journal of Computer Research and Development, 2003.

[13] Li Liang Ding Wanshan, "Paper currency number recognition system based on ARM", Electronic Measurement Technology;2008-10.

[14] Jin Ye, Liu Songbo, Liu Jiafeng, Song Ling, and Tang Xianglong, "An Edge-Based Defect Detection Algorithm for Paper Currency", Journal of Computer Research and Development, 2007-02.

[15] K. Fanhui, M. Jiquan, G. Xin, Y. Liping, "Paper Currency Recognition Using Gaussian Mixture Models Based on Structural Risk Minimization", Journal of Computer Engineering and Applications, 2006-13.

[16] Y. Weiqi, Z. Yu, "A Fast Recognition System for Paper Currency Numbers" , Journal of Computer Engineering, 2005-24.

[17] M. M. Alimushwan, A. Mohaimin, R. Islam, S. Chowdhury, M. H. Ali, "Fake Currency Detection using Image Processing Method", BRAC university dspace.

[18] Z. Ahmed, S. Yasmin, M. N. Islam, R. U. Ahmed, "Image Processing Based Feature Extraction of Bangladeshi Banknotes", The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014), 18-20 Dec. 2014.

[19] M. M. Rahman, Bruce Poon, M. A. Amin, H. Yan, "Recognizing Bangladeshi Currency for Visually Impaired", International Conference on Communication in Computer and Information Science, July 2014.

[20] R. F. Sajal, M. Kamruzzaman, F. A. Jewel, "A machine vision based automatic system for real time recognition and sorting of Bangladeshi bank notes", 2008 11th International Conference on Computer and Information Technology (ICCIT), 24-27 December, 2008.

[21] K. K. Debnath, J. K. Ahdikary, M. Shahjahan, "A currency recognition system using negatively correlated neural network ensemble", 2009 12th International Conference on Computer and Information Technology (ICCIT), 21-23 December, 2009.

[22] K.Yoshida, M. Kamruzzaman, F. A. Jewel, R. F. Sajal, "Design and implementation of a machine vision based but low cost stand alone system for real time counterfeit Bangladeshi bank notes detection", 2007 10th International Conference on Computer and Information Technology (ICCIT), 27-29 December, 2007.

[23] Demonstration Video Link: http://bit.ly/30CNxWz

An Analytical Approach for Enhancing the Automatic Detection and Recognition of Skewed Bangla License Plates

Koushik Roy*†, Abu Mohammad Shabbir Khan*†,
Mohammad Zariff Ahsham Ali†, Sazid Rahman Simanto†, Nabeel Mohammed†,
Muhammad Asif Atick‡, Shahidul Islam‡, Kazi Mejbaul Islam‡
†Department of Electrical and Computer Engineering, North South University
Email: {koushik.roy, abu.khan1, zariff.ahsham, sazid.rahman, nabeel.mohammed}@northsouth.edu
‡HeadBlocks
Email: {asif, shahidul, kazimejbaul}@head-blocks.com

***Abstract*—Although there has been a huge body of work on Bangla license plate detection and recognition, the successes of these works have largely been limited to correct detection and recognition of undistorted license plates whose images are taken chiefly from the front or the back of vehicles with slight angular variations. As a result, most Bangla automatic license plate recognition (ALPR) systems in practice struggle when the license plates are skewed on the viewing or the image planes of the license plates. In this paper, we address this issue by proposing an analytical approach that can enhance the ALPR of both normal and skewed license plates and can be incorporated into existing Bangla ALPR systems without modifying their internal structures. Specifically, we demonstrate how existing ALPR systems can be treated as black boxes and analyzed to understand what sort of license plate images they work best on and introduce a novel pipeline that combines deep learning and an algorithmic procedure for transforming images of both normal and skewed license plates into formats that are best suited for the ALPR systems. We note that our proposed method can be easily generalized and applied to non-Bangla license plates as well.**

***Index Terms*—Bangla, Automatic License Plate Recognition, Deep Learning**

I. Introduction

Correct recognition of vehicle license plates has numerous use cases that include penalizing irresponsible driving and parking, keeping track of vehicles coming in and going out of parking lots, identifying vehicle ownership and so on. While a lot of work on automatic detection and recognition of license plates has been done for decades, there is still a lot of progress to be made on correct Bangla automatic license plate recognition (ALPR). The variance that exists in Bangla license plates owing to existence of different metro and vehicle types, and the scarcity of adequate samples to catch all of that variety makes Bangla ALPR specially challenging. A Bangladeshi company named HeadBlocks is working with the Dhaka Metropolitan Police (DMP) to address this and has developed a commercial Bangla ALPR system with a four stage pipeline that includes license plate detection, character segmentation, character detection, and character recognition. Although the ALPR system developed by HeadBlocks (referred to as HB-ALPR henceforth) has over 96% accuracy as determined by experiments on their confidential test set, HB-ALPR finds it difficult to correctly identify license plates that are more than 30° skewed on the image plane (*ip-skewed*) or the viewing plane (*vp-skewed*). Examples of such *ip-skewed* and *vp-skewed* license plates are shown in Fig. 1.

Since HB-ALPR has been commercially deployed and cannot be modified without affecting all four stages of its pipeline and extensive testing, we approached the problem of correctly identifying skewed license plates by treating HB-ALPR as a black box. Concretely, we first analyzed what kind of license plate images are ideal for HB-ALPR, and then by combining a deep learning method with an algorithmic procedure, we developed a pre-processing step for the license plate images that led to a performance improvement of HB-ALPR for both skewed and challenging non-skewed images.

The rest of the work is organized as follows. In Section II, we describe some related work done on Bangla and non-Bangla ALPR. In Section III, we describe our data set, the analysis we did to understand the strengths and weaknesses of HB-ALPR, and the method we developed to pre-process the images to the best format for HB-ALPR. In Section IV, we show how our method affected the performance of HB-ALPR, and then in Section V, we present our conclusions and avenues of future work.

II. Related Work

While there have been a number of works on Bangla ALPR based on algorithmic approaches such as [1] (which focused on license plate detection only) and [2] (which used template matching), shallow machine learning methods such as [3] (which used support vector machines), and deep learning such as [4] (which used convolutional neural networks), none of these works present extensive evaluations of their performances on skewed license plate images. Moreover, to the best of our knowledge, no work has been done so far to augment

* denotes equal contribution.

(a) IP-Skewed Ex. I

(b) IP-Skewed Ex. II

(c) VP-Skewed Ex. I

(d) VP-Skewed Ex. II

Fig. 1. Examples of Different Types of Skewed License Plate Images

the performances of existing Bangla ALPR systems through pre-processing their input images while treating the systems as black boxes.

As in the case of Bangla ALPR systems, most ALPR systems developed for non-Bangla license plates such as [5], [6], [7], and [8] do not demonstrate good performances or show thorough assessments on skewed license plates.

Silva and Jung [9], however, focused on building a complete ALPR system robust to skewness of license plates which comprised of three main steps: vehicle detection, license plate detection and unwarping, and optical character recognition (OCR). In this work, they introduced a novel network named Warped Planar Object Detection Network (WPOD-NET) that searches for license plates in the region of each detected vehicle and computes parameters for an affine transformation that enables the rectification of each detected license plate to a rectangular frontal view. Our proposed solution is similar to WPOD-NET in that it too produces a rectangular frontal view of the detected license plate, but it does so in a different way. In addition, our proposed approach is applicable to any ALPR system and is not confined to any specific pipeline as WPOD-NET is.

III. Methodology

A. The Dataset

Our dataset consisted of around 3000 license plate images of varying sizes taken from different perspectives using different cameras. For bringing uniformity, we rescaled all images to 400×400 because the character segmentation and recognition networks of HB-ALPR were trained using images of that size. While resizing images, we opted to keep their original aspect ratios the same and added black padding where it was necessary. An example set of such images after converting them to the size 400×400 is shown in Fig. 2.

We then used three annotators to label the images as *vp-skewed*, *ip-skewed*, or *normal* (all being mutually exclusive) and used the majority vote of the annotators to decide the final label of each image. Next, we randomly selected 1015 images and created mask annotations of the license plates in these images using a tool called VIA [10].

B. The Proposed Solution

Our proposed method of pre-processing license plate images consists of two steps: performing instance segmentation of license plates on images and then transforming the segmented license plates into uniform rectangular views. The details of each of the steps are provided in the following sub-subsections.

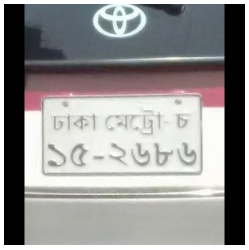
1) Instance Segmentation of License Plates: We conducted transfer learning on an existing Mask R-CNN [11] model and fine tuned it for Bangla license plates to determine exactly which pixels a license plate consisted of and to regress the smallest rectangular bounding box that contained the entirety of the license plate.

To train our Mask R-CNN, we divided the 1015 mask annotated license plate images into a training set of 800 images and a validation set of 215 images. By selecting a learning rate of 0.001 and loading 2 images to an NVIDIA K80 GPU at a time, we trained our model for 100 epochs with the batch size set equal to 100. For the rest of the hyperparameters, we used the same values as mentioned in [11]. The outputs of our Mask R-CNN model are shown in Fig. 3.

2) Perspective Transformation of License Plates: Using the predicted masks from our Mask R-CNN model, we generated the *Shi-Tomasi Corners* [12] reconstructed from *Harris Corners* [13] of each license plate instance. Through our empirical studies, we found that padding the mask by 10 pixels led to

(a) 400x400

(b) 600x600



(c) 332x400

(d) 3840x2160

Fig. 2. Images of Different Sizes after Conversion to the Size 400x400

(a) IP-Skewed Ex. I (b) IP-Skewed Ex. II (c) VP-Skewed Ex. I (d) VP-Skewed Ex. II

Fig. 3. Instance Segmentation of the License Plates in Fig. 1 Using Our Mask R-CNN Model (Best Viewed in Color)

slight improvement in corner detection.

Next, for each detected license plate, we put its *Shi-Tomasi Corners* in an array of tuples $Points$ where in each tuple, the first element ($Points[i]_x$) represented the x co-ordinate and the second element ($Points[i]_y$) represented the y co-ordinate of a *Shi-Tomasi Corner* i.

$$S = Points[i]_x + Points[i]_y; \forall i \tag{1}$$

$$D = Points[i]_x - Points[i]_y; \forall i \tag{2}$$

We then constructed the vectors S and D that respectively store the summations of and differences between the x and y co-ordinates of the *Shi-Tomasi Corners*.

$$topLeft_x, topLeft_y = Points[argmin(S)] \tag{3}$$

$$topRight_x, topRight_y = Points[argmax(D)] \tag{4}$$

$$bottomLeft_x, bottomLeft_y = Points[argmin(D)] \tag{5}$$

$$bottomRight_x, bottomRight_y = Points[argmax(S)] \tag{6}$$

After that, we used (3), (4), (5), and (6) to determine the four extreme corner points—$topLeft$, $topRight$, $bottomLeft$, and $bottomRight$—of each detected license plate and used these points to project the detected license plate to a rectangular frontal view. Subsections III-C and III-D describe how we decided the dimensions of the rectangle. For now, assume that each extreme corner j was mapped to a new rectangle corner $j^{'}$. We computed the 3×3 map matrix M for the perspective transformation using (7).

$$\begin{bmatrix} j_x^{'} & j_y^{'} & 1 \end{bmatrix} = \begin{bmatrix} j_x & j_y & 1 \end{bmatrix} \times M \tag{7}$$

Finally, we warped each detected license plate in to the image out on a normal viewing plane using (8) where $out(x, y)$ denotes the pixel value placed at the (x, y) position of out.

$$out(x, y) = in\Big(f_1(x, y), f_2(x, y)\Big) \tag{8}$$

where

$$f_1(x, y) = \frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}} \tag{9}$$

$$f_2(x, y) = \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}} \tag{10}$$

C. Analysis of HB-ALPR

To determine what is the ideal size of license plate images for HB-ALPR, we took 15 samples each from the sets of originally 400×400, 600×600, 1280×720, and 3840×2160 sized images and 29 samples from images of different sizes. The samples were a mixture of apparently good looking and somewhat skewed and distorted images. Note that despite having different original sizes, all the samples were resized to 400×400 pixels.

We then found the boxes of the license plates in these images using our Mask R-CNN model and produced two different types of outputs: one where each pixel outside the license plate bounding box was made black (*black bbox*) and one where each such pixel was made white (*white bbox*). At this point, we had three representations of each image: the original one (*org original*), the *black bbox* image, and the *white bbox* image where each representation had the dimensions 400×400. As HB-ALPR uses a fully convolutional network in the first stage of its pipeline, we further resized each of these representations to the size 200×200 to see how HB-ALPR performs for small images. Next, we assessed how the accuracy (determined by for what fraction of license plates *all the digits* on each plate were identified correctly) of HB-ALPR varied for these six representations using BGR and RGB color schemes as shown in Fig. 4 where each bar presents metrics for each set of 89 samples.

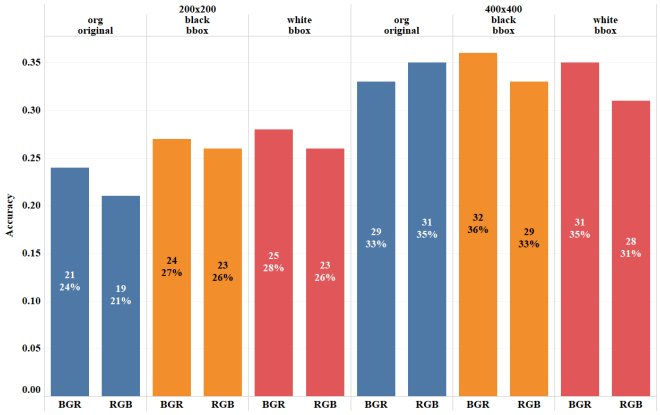


Fig. 4. Accuracy Distribution for Mixture of Good and Challenging Images

(a) IP-Skewed Ex. I

(b) IP-Skewed Ex. II

(c) VP-Skewed Ex. I

(d) VP-Skewed Ex. II

Fig. 5. The Deskewed Versions of the License Plates in Fig. 1

D. The Final Output

From our analysis, we noticed that the license plates that occupied a large portion of the images and were causing HB-ALPR to fail in the 400×400 case, were being correctly recognized after their images were converted to the size 200×200. Moreover, the license plates that seemed to have ideal dimensions in the 400×400 case were being incorrectly identified by HB-ALPR in the 200×200 case because they became too small. This meant that the performance of HB-ALPR was being ultimately affected by the size of the license plates in the images instead of the size of the images. As we had bounding box co-ordinates of the license plates owing to our Mask R-CNN model, we used those co-ordinates to calculate the size of the license plates in our samples and found that the best performance was being obtained when the license plates had dimensions in the ballpark of 150×75 pixels. Also, our analysis showed that the black padding had a positive effect on license plate recognition despite HB-ALPR not being trained with such images. Lastly, we found that the RGB color scheme worked best in most of the cases.

So we decided to deskew the detected license plates to the size 150×75 pixels and then to add black padding to make the entire image of size 400×400 pixels. The deskewed versions of the images in Fig. 1 are shown in Fig. 5.

IV. Results

To understand the impact of our work, we constructed a test dataset with 25 *challenging normal* images, 25 *ip-skewed* images, and 25 *vp-skewed* images for a total of 75 images.

As shown in Table I, the recognition accuracy for *challenging normal*, *ip-skewed*, and *vp-skewed* images went up by 45%, 86%, and 25% respectively after our pre-processing work. In addition, the detection % of license plates (defined by the percentage of license plates detected successfully) of HL-ALPR became almost perfect.

V. Conclusion and Future Work

By pre-processing the input images of Bangla ALPR systems, we have demonstrated how the performance of such systems can be improved for both normal and skewed images despite treating the systems as black boxes. We are currently working on other license plate distortions such as blurriness, fading, low illumination, occlusion et cetera and would be soon publishing the results of our work for those cases.

TABLE I
Impact of Deskewing on HB-ALPR

Type of Image	Accuracy Distribution		Detection % Distribution	
	Original	Deskewed	Original	Deskewed
challenging normal	44%	64%	84%	100%
ip_skewed	28%	52%	92%	100%
vp_skewed	48%	60%	88%	96%

References

[1] S. Azam and M. M. Islam, "Automatic license plate detection in hazardous condition," *Journal of Visual Communication and Image Representation*, vol. 36, pp. 172–186, 2016.

[2] A. C. Roy, M. K. Hossen, and D. Nag, "License plate detection and character recognition system for commercial vehicles based on morphological approach and template matching," in *2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*. IEEE, 2016, pp. 1–6.

[3] M. A. Uddin, J. B. Joolee, and S. A. Chowdhury, "Bangladeshi vehicle digital license plate recognition for metropolitan cities using support vector machine," in *Proc. International Conference on Advanced Information and Communication Technology*, 2016.

[4] M. Z. Abedin, A. C. Nath, P. Dhar, K. Deb, and M. S. Hossain, "License plate recognition system based on contour properties and deep learning model," in *2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*. IEEE, 2017, pp. 590–593.

[5] R. Laroca, E. Severo, L. A. Zanlorensi, L. S. Oliveira, G. R. Gonçalves, W. R. Schwartz, and D. Menotti, "A robust real-time automatic license plate recognition based on the yolo detector," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–10.

[6] S. M. Silva and C. R. Jung, "Real-time brazilian license plate detection and recognition using deep convolutional neural networks," in *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 2017, pp. 55–62.

[7] Z. Selmi, M. B. Halima, and A. M. Alimi, "Deep learning system for automatic license plate detection and recognition," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 1132–1138.

[8] H. Li, P. Wang, and C. Shen, "Towards end-to-end car license plates detection and recognition with deep neural networks," *ArXiv*, vol. abs/1709.08828, 2017.

[9] S. Montazzolli Silva and C. Rosito Jung, "License plate detection and recognition in unconstrained scenarios," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 580–596.

[10] A. Dutta, A. Gupta, and A. Zissermann, "VGG image annotator (VIA)," http://www.robots.ox.ac.uk/ vgg/software/via/, 2016, version: 2.0.6, Accessed: Mar. 20, 2019.

[11] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[12] J. Shi and C. Tomasi, "Good features to track," Cornell University, Tech. Rep., 1993.

[13] C. G. Harris, M. Stephens *et al.*, "A combined corner and edge detector." Citeseer, 1988.

AIBangla: A Benchmark Dataset for Isolated Bangla Handwritten Basic and Compound Character Recognition

Md Mahedi Hasan*, Mahathir Mohammad Abir†, Mohammad Ibrahim†, Muhammad Sayem† and Sohaib Abdullah†
*Institute of Information and Communication Technology (IICT)
Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh
†Department of Computer Science and Engineering
Manarat International University, Dhaka, Bangladesh

***Abstract*—Automatic handwritten Bangla character recognition (HBCR) is a challenging problem in computer vision due to numerous variations in writing styles of an individual Bangla character and the presence of similarities in shapes among different characters. Considering the complexity of the problem, we need to develop a modern convolutional neural network (CNN) for accurate recognition, but unfortunately, at present, very few Bangla handwritten dataset contain a large number of image samples for each character suitable for training deep learning-based methods. In this paper, we present AIBangla, a new benchmark image database for isolated handwritten Bangla characters with detailed usage and a performance baseline. Our dataset contains 80,403 hand-written images on 50 Bangla basic characters and 249,911 hand-written images on 171 Bangla compound characters which were written by more than 2,000 unique writers from various institutes across Bangladesh. In addition, we have applied three leading state-of-the-art deep CNN networks on our proposed AIBangla dataset to provide baseline performance. We have achieved a maximum accuracy of 98.13% and 81.83% for basic and compound character classes respectively on the test set of the AIBangla dataset.**

***Index Terms*—OCR, Bangla Compound character, Image classification, Benchmark database, Handwritten character recognition**

I. Introduction

Bangla, an Indo-Aryan language, is the national and official language of Bangladesh. It is ranked as the sixth most spoken language in terms of number of native speakers. The Bangla characters have been used to write Sanskrit-based script. In addition to 50 basic character classes, as per different enumerations of experts, there are around 334 complex shaped compound characters in Bengali script out of which 171 classes are more frequent than others [1]. These complex shaped characters also have enormous variations in their writing styles where some different characters have close resemblance with one another making automatic handwritten Bangla character recognition (HBCR), the first step of many important applications like optical character recognition (OCR), word recognition etc., a difficult task. Additionally, the variation in size and shape of handwritten characters among different writers makes it more difficult to recognize, as compared to the printed forms of the character.

In recent years, due to the powerful feature learning abilities of deep neural networks, deep learning-based methods have achieved promising performance in many computer vision tasks. The success of deep learning-based methods greatly depends on the vast number of labeled data. Unfortunately, only few existing HBCR datasets contain a large number of images covering all available basic and compound character classes.

In this paper, a new challenging dataset for Bangla isolated handwritten character recognition is introduced. The dataset has been prepared from handwritten samples collected from various institutes throughout Bangladesh and checked through a rigorous procedure to ensure accuracy. This dataset contains more than $330,000$ images of 221 standard character classes which can be used for both handwritten basic and compound character recognition in using deep learning-based approaches.

The prime contributions of our paper are as follows:

1) We have introduced a new challenging dataset for Bangla isolated handwritten character recognition which contains more than $330,000$ images in 221 different Bangla characters comprising of 50 basic and 171 frequently used compound characters collected from students of different ages and genders from various institutes of Bangladesh. The dataset is made publicly available which can found at this link http://dx.doi.org/10.17632/hf2tt9kxkn.1

2) Our AIBangla dataset is the largest in terms of the number of compound character classes and image samples compared to other publicly available HBCR datasets as shown in Table I.

3) In order to provide baseline, we have also applied several state-of-the-art deep learning-based algorithms on our AIBangla dataset and achieved comparable performance with other prevalent methods.

The rest of the paper is organized as follows: section II discusses existing dataset for Bangla handwritten character recognition, while section III presents our proposed AIBangla dataset with complete description of data acquisition and extraction procedures. Baseline recognition is reported in section IV. Finally, we summarize our results in section V

978-1-7281-5242-4/19

Fig. 1. Sample images of Bangla alphabets of our proposed AIBangla dataset. The dataset contains more than $80,000$ images on 50 basic character classes with on average $1,608$ images in per alphabet class.

along with the directions for future work.

II. Existing Datasets

Several works have been carried out for handwritten Bangla character recognition (HBCR) over the last three decades. First effective work in HBCR was done by Ray and Chatterjee [2]. After that, many methods have been studied [3]–[5] to improve the performance of recognition in order to build a complete Bangla OCR system. Dutta et al. [3] developed a two-stage feed-forward neural network based recognition scheme to recognize multifont alpha-numeric Bangla characters. Bhattacharya et al. [4] trained a multilayer perceptron (MLP) network by the back-propagation algorithm to classify 50 basic Bangla alphabets where features were obtained by computing local chain code histograms of the input character shape. In [6] Hasnat et al., proposed a technique where discrete cosine transform (DCT) was applied over the input image and Hidden Markov Model (HMM) was used for both printed and handwritten character recognition.

The above-mentioned methods, however, used many handcrafted features extracted from a small dataset which turned out to be less effective for modern deep learning-based methods. The success of deep learning-based methods greatly depends on the size of the training data. Therefore, to develop a robust HBCR, training database should be large-scale containing a large number of images per class. Some of the publicly available HBCR databases are ISI datasets [7], CMATERdb dataset [1], BanglaLekha-Isolated dataset [8] and Ekush dataset [9]. Sample images of these datasets are shown in Fig. 2, whereas, Table I presents a comparison based on several different parameters among different publicly available datasets for Bangla handwritten character recognition.

Fig. 2. Sample images of different publicly available datasets for isolated Bangla handwritten character recognition. (a) ISI (b) CMATERdb 3.1.3.3 (c) BanglaLekha-Isolated (d) Ekush dataset

ISI datasets [7] have a total $37,858$ images for isolated Bangla alphabets and $23,299$ images for Bangla numerals. But, unfortunately, the dataset only contains images of these two categories and does not have any standard Bangla compound character.

CMATERdb 3.1.3.3 dataset [1] contains images of a large compound character classes. It contains total $58,662$ images from 221 classes out of which 171 classes are compound. That makes the average number of samples per character class is around 277, which is not large enough for modern deep learning-based methods.

The BanglaLekha-Isolated dataset [8] has a total $166,105$ handwritten images of 84 classes (50 basic, 10 numerals and 24 compound characters). Presence of a few numbers of compound character classes make it less useful for the comprehensive character recognition task.

The Ekush dataset [9] is the largest dataset for Bangla handwritten character recognition which consists of $367,018$ images in total 122 classes in four different categories. Unfortunately, the dataset contains sample images of only 52 compound characters making it less useful for the comprehensive character recognition task. Moreover, it contains online handwritten characters which are substantially different from offline handwritten characters.

III. AIBangla Dataset Preparation

In this section, the steps regarding data acquisition, characters extraction, data preprocessing and error correction methodologies of the AIBangla dataset are briefly described.

A. Data acquisition

The AIBangla dataset was collected from students across various institutes of Bangladesh. Students were given data collection forms to write down the basic and compound characters of Bangla language. Each form was designed to write one specific basic or compound character. It had a regular grid pattern in which the characters were written. The filled-in collection forms were then scanned at 300 d.p.i. resolution using HP Scanjet scanners. In this version of the dataset, only isolated handwritten character images were collected. Figure 3 shows a sample data collection form which was used to collect handwriting samples. Students were asked to write one given Bangla character printed on top of the form at their natural

TABLE I
COMPARISON AMONG DIFFERENT PUBLICLY AVAILABLE DATASETS FOR BANGLA HANDWRITTEN CHARACTER RECOGNITION.

Type	Basic Characters		Compound Characters		Numeral		Total	
Dataset	Class	Samples	Class	Samples	Class	Samples	Class	Samples
ISI [7]	50	37,858	None		10	23,299	60	61,157
CMATERdb 3.1.3.3 [1]	50	15,103	171	42,959	10	6,000	231	58,662
BanglaLekha-Isolated [8]	50	98,950	24	47,407	10	19,748	84	166,105
Ekush [9]	50	154,824	52	150,840	10	30,687	112	336,351
AIBangla	50	80,403	171	249,911	None		221	330,314

TABLE II
DATA COLLECTION STATISTICS OF OUR PROPOSED AIBANGLA DATASET. DUE TO THE RIGOROUS MANUAL SCREENING, 15% OF THE SAMPLES WERE DISCARDED. AVERAGE SAMPLE IMAGES PER CLASS ARE AROUND 1,500, WHICH WOULD BE SUFFICIENT FOR TRAINING DEEP NEURAL NETWORK-BASED ARCHITECTURES FOR HBCR.

Dataset type	Classes	Form per character	Extracted samples	Discarded samples	Total samples	Avg. per class	Standard deviation
Basic	50	15	94,250	13,847	80,403	1,608	50.07
Compound	171	13	288,990	39,079	249,911	1,461	123.23
Total	221	-	383,240	52,926	330,314	1,495	106.68

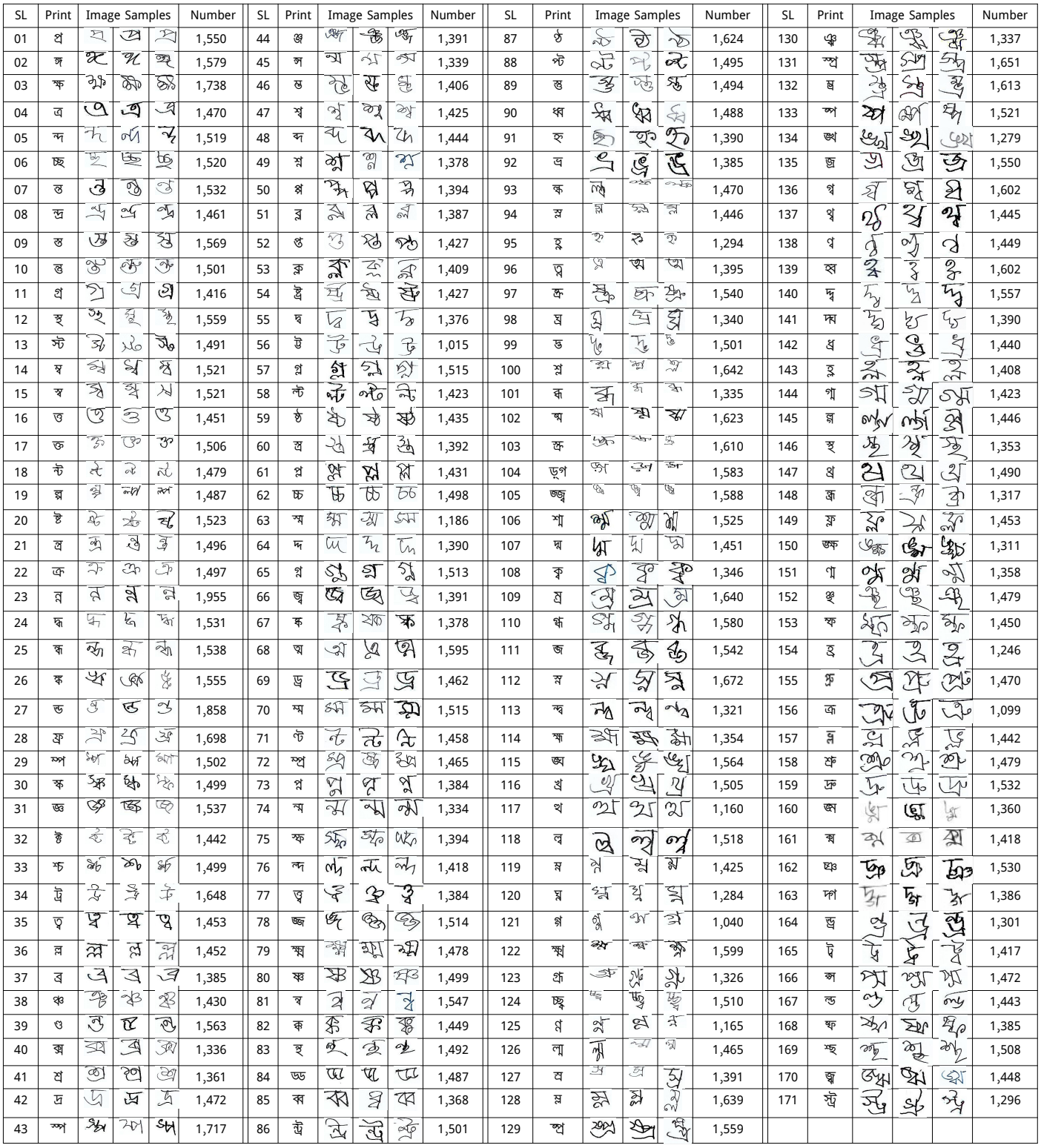
Fig. 3. Sample scan image of a filled data collection form of our proposed AIBangla dataset. Each form was designed to write one specific character printed on top of the form. A completely filled-in form contains a total of 130 character samples in its 13 rows and 10 columns.

style. Table II illustrates data collection statistics of our dataset in detail.

B. Data Extraction

Bangla characters were being extracted from scanned images using following procedure:

- *Binarization:* Our first step was to convert the acquired scanned image into the grayscale format. Adaptive thresholding method was used to further convert the grayscale image into a binary image in order to reduce the illumination variation and background noise. To accomplish this, a local thresholding method that considers local contrast by using local descriptors, for example, mean and the standard deviation was employed.
- *Contour and corner detection:* The next step was to identify the four corners of the grid which corresponds to the detection of the largest quadrilateral box in the image. We determined the largest outer box by calculating the area enclosed by each connected component and then identifying the largest value. The intersection of the four lines gave the corners of the grid. After that, the extracted box was used to generate a binary mask to eliminate all the excessive elements of the image.
- *Geometric transformation:* Since the images were scanned manually they can be misaligned or tilted at some random angle. In such cases, to estimate the deflected-angle of the image, the Hough transform was used. Additionally, geometric transformation of the image was done to account for any kind of perspective distortion.
- *Segmenting cells:* In this step, we had to segment 130 characters from 130 cells of the scanned image containing 13 rows and 10 columns. To accomplish this, first we



SL	Print	Image Samples	Number	SL	Print	Image Samples	Number	SL	Print	Image Samples	Number	SL	Print	Image Samples	Number
01	প্র		1,550	44	জ্ঞ		1,391	87	ষ্ঠ		1,624	130	ঞ্চ		1,337
02	ঙ্গ		1,579	45	ন্স		1,339	88	ষ্ট		1,495	131	স্প্র		1,651
03	ক্ষ		1,738	46	ম্ভ		1,406	89	স্ত		1,494	132	ম্ব্র		1,613
04	ত্র		1,470	47	শ্ব		1,425	90	ধ্ব		1,488	133	শ্প		1,521
05	ন্দ		1,519	48	ব্দ		1,444	91	হ্ন		1,390	134	জ্ঞ		1,279
06	চ্ছ		1,520	49	শ্ন		1,378	92	ভ্র		1,385	135	জ্ঞ		1,550
07	ন্ত		1,532	50	প্প		1,394	93	ল্ক		1,470	136	থ্ব		1,602
08	ন্দ্র		1,461	51	ন্ন		1,387	94	ম্ম		1,446	137	থ্ব		1,445
09	ন্ত		1,569	52	ণ্ড		1,427	95	হ্ম		1,294	138	ণ্ব		1,449
10	ন্ত		1,501	53	ক্ল		1,409	96	ত্র		1,395	139	ষ্ব		1,602
11	প্র		1,416	54	ষ্ট্র		1,427	97	ক্ত		1,540	140	দ্ব		1,557
12	স্থ		1,559	55	ব্ব		1,376	98	ম্র		1,340	141	দ্ম		1,390
13	স্ট		1,491	56	ড্ড		1,015	99	ভ্ত		1,501	142	ধ্র		1,440
14	ষ্ব		1,521	57	গ্ল		1,515	100	ম্প		1,642	143	হ্ল		1,408
15	শ্ব		1,521	58	ল্ট		1,423	101	ব্ধ		1,335	144	গ্ম		1,423
16	ত্ত		1,451	59	ষ্ঠ		1,435	102	শ্ব		1,623	145	ল্ল		1,446
17	ক্ত		1,506	60	ত্র		1,392	103	ক্ল		1,610	146	হ্ম		1,353
18	ল্ট		1,479	61	প্ল		1,431	104	ড়্গ		1,583	147	থ্র		1,490
19	ল্ল		1,487	62	চ্চ		1,498	105	জ্ঞ		1,588	148	ক্র		1,317
20	ষ্ট		1,523	63	শ্ম		1,186	106	শ্ম		1,525	149	ফ্র		1,453
21	ন্ত্র		1,496	64	দ্দ		1,390	107	দ্ম		1,451	150	ঙ্ক্ষ		1,311
22	ক্র		1,497	65	গ্র		1,513	108	ক্ব		1,346	151	গ্ন		1,358
23	ন্ন		1,955	66	জ্ঞ		1,391	109	ম্র		1,640	152	ঞ্ছ		1,479
24	দ্ধ		1,531	67	ক্ক		1,378	110	ধ্ম		1,580	153	ক্ষ		1,450
25	ব্ধ		1,538	68	অ্য		1,595	111	জ্ঞ		1,542	154	হ্ন		1,246
26	ক্ষ		1,555	69	ভ্র		1,462	112	ম্ম		1,672	155	ষ্ণ		1,470
27	ভ্র		1,858	70	ম্ম		1,515	113	ন্ধ		1,321	156	ক্র		1,099
28	ফ্র		1,698	71	ণ্ট		1,458	114	ঙ্ক্ষ		1,354	157	ন্ন		1,442
29	স্প		1,502	72	স্প্র		1,465	115	জ্ঞ		1,564	158	শ্চ		1,479
30	ক্ষ		1,499	73	প্প		1,384	116	থ্র		1,505	159	দ্ভ		1,532
31	জ্ঞ		1,537	74	ন্ম		1,334	117	থ্র		1,160	160	জ্ঞ		1,360
32	ঙ্ক		1,442	75	ক্ষ		1,394	118	ন্ব		1,518	161	ঋ		1,418
33	চ্চ		1,499	76	ল্দ		1,418	119	ম্ন		1,425	162	ঞ্ছ		1,530
34	ট্র		1,648	77	ত্ত্ব		1,384	120	ম্ম		1,284	163	দ্গ		1,386
35	ত্ব		1,453	78	জ্জ		1,514	121	গ্ন		1,040	164	দ্র		1,301
36	ন্ন		1,452	79	ক্ষ্ম		1,478	122	ক্ষ্ম		1,599	165	দ্ব		1,417
37	ব্র		1,385	80	ঙ্ক্ষ		1,499	123	ঞ্চ		1,326	166	ল্ল		1,472
38	ঞ্চ		1,430	81	ব্ব		1,547	124	ঙ্ক্ষ		1,510	167	ল্ভ		1,443
39	ণ্ড		1,563	82	ক্ক		1,449	125	ন্ন		1,165	168	ঙ্ক		1,385
40	ক্স		1,336	83	হ্ম		1,492	126	ল্ম		1,465	169	ঙ্ছ		1,508
41	শ্র		1,361	84	ড্ড		1,487	127	ম্ন		1,391	170	জ্ঞ		1,448
42	দ্র		1,472	85	ব্ব		1,368	128	ম্ম		1,639	171	ষ্ট্র		1,296
43	শ্ম		1,717	86	ষ্ট্র		1,501	129	প্র		1,559				

Fig. 4. Sample images of Bangla compound characters of our proposed AIBangla dataset. The dataset contains more than $24,9911$ images on 171 compound character classes with on average $1,461$ images in per character.

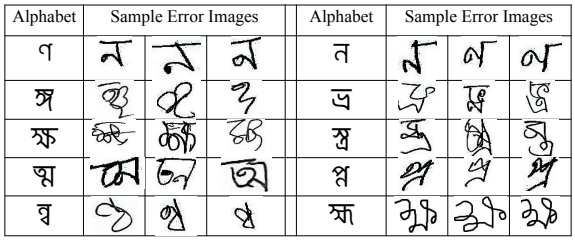


Fig. 5. Sample images which were discarded during manual screening. 15% of the total collected samples were removed in this process. Discarded characters were either misclassified or were not properly written.

identified the coordinates of 130 cells from adjacent pairs of vertical and horizontal grid lines. These coordinates were then used to crop the original grayscale image.

C. Error Correction

The extracted character images have been undergone a rigorous manual screening in order to ensure an error-free dataset. The following steps illustrate the manual screening:

- Firstly, we carefully examined the extracted images found after data extraction procedure from scanned images described in III-B. During this stage, characters which were improperly extracted or blank or misclassified were removed.
- After that, these extracted images were grouped into 221 folders according to their classes. Images from each of these classes were then re-examined by at least three authors of this paper. Characters which were not properly written were discarded in this step.
- The entire dataset was evaluated again to discard erroneous images which were not filtered out in the previous two steps.

The above-mentioned rigorous methodologies of manual screening will definitely ensure dataset quality and correctness of the labels. Some of the sample discarded images of the AIBangla dataset are shown in Fig. 5.

IV. Character Recognition Baseline

In this section, we describe three leading state-of-the-art deep convolutional neural network (DCNN) architectures that were used as a baseline for basic and compound character recognition on our AIBangla dataset. The section also illustrates the training steps and the performance evaluation procedures of these networks.

A. Experimental Setup

To evaluate the performance of character recognition we conducted two experiments on the AIBangla dataset. The first experiment we performed on this dataset was to recognize the basic 50 alphabets of Bangla language which includes 11 vowels and 39 consonants. Figure 1 shows the sample images of those 50 basic characters of the AIBangla dataset. In our second experiment, we performed recognition of 171 compound characters available in AIBangla dataset. Figure 4 shows sample images of those 171 compound characters. The dataset was divided into a training and a test set. 80% random images of each character class were kept for training and 10% images were kept for validation purposes. Table III shows the experimental setup.

TABLE III
Experimental setup for basic and compound character recognition on AIBangla dataset. Total 80% of the random sample images were used for training, and for validation and test purposes 10% images were set. Experiments on basic character recognition were performed on 50 Bangla alphabets and compound character recognition was performed on 171 available compound characters on AIBangla dataset.

Experiment	Total Class	Total Samples	Train. Samples	Val. Samples	Test Samples
Basic	50	80,403	64,300	8,053	8,050
Compound	171	249,911	199,900	25,006	25,005

1) Training: In this experiment, we used the three most popular state-of-the-art deep convolutional neural network (DCNN) architectures VGG-16 [10], Resnet-50 [11], and DenseNet [12] to provide baseline character recognition on AIBangla dataset. These models have been pretrained with ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [13] dataset. We employed Transfer Learning scheme to use this pretrained model. The last fully connected (FC) layer has been replaced by a softmax layer of 50 neurons to classify 50 classes for basic character recognition and by a softmax layer of 171 neurons to classify 171 classes for compound character recognition.

All these deep CNN architectures were trained using Adam [14] optimization algorithm for 200 epochs with a batch size of 64. We used categorical cross-entropy as our loss function. The initial learning rate was set to 10^{-3} and we reduced it gradually as the training progressed. All the input images were resized to 224 x 224, with keeping their aspect-ratio by adding gray-scale padding. We did not use any data augmentation technique to increase the number of train samples.

2) Performance Evaluation: Among the three state-of-the-art DCNN networks, DenseNet [12] shows the best performance on the test set of the proposed AIBangla dataset for both isolated handwritten Bangla basic and compound character recognition experiment. Table IV shows the comparison of these networks for both experiments. Figure 6 illustrates the same comparison with bar chart.

V. Conclusions

In this paper, a new challenging benchmark dataset for Bangla isolated handwritten character recognition has been introduced. This dataset is mainly intended for the comprehensive development of Bangla handwritten character recognition using modern deep learning-based techniques. In contrast to existing datasets, our AIBangla dataset provides a large number of images in all available handwritten basic and compound characters of Bangla language with a variety of

TABLE IV

COMPARISON AMONG DIFFERENT STATE-OF-THE-ART DCNN IN ISOLATED HANDWRITTEN BANGLA BASIC AND COMPOUND CHARACTER RECOGNITION ON THE TEST SET OF PROPOSED AIBANGLA DATASET. FROM THE COMPARISON, IT IS SEEN THAT AMONG ALL OTHER NETWORKS DENSENET [12] PROVIDES THE BEST PERFORMANCE ON BOTH BASIC AND COMPOUND HANDWRITTEN CHARACTER SET.

Methods	Basic-50 Character Set	Compound-171 Character Set
VGG-16 [10]	96.75	77.82
Resnet-50 [11]	97.02	79.23
DenseNet [12]	**98.13**	**81.83**

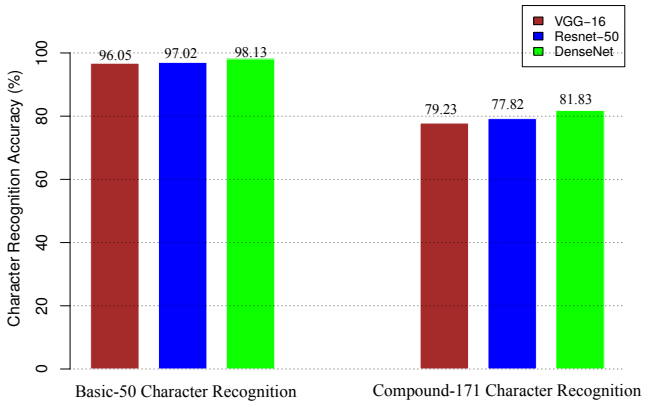


Fig. 6. Comparison among different state-of-the-art DCNN methods in both basic and compound character recognition accuracy on the test set of AIBangla dataset. DenseNet achieves the best recognition accuracy in both of the experiment.

variations. This dataset will surely help researchers to develop handwritten OCR-based applications in one of the widely used scripts of the world. We also present a baseline character recognition network for this dataset which shows that the dataset is challenging for current state-of-the-art character recognition methods.

We will, in the future, intend to add more images in our present dataset including Bangla numerals and modifiers from locations.

REFERENCES

[1] N. Das, K. Acharya, R. Sarkar, S. Basu, M. Kundu, and M. Nasipuri, "A benchmark image database of isolated Bangla handwritten compound characters." *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 17, no. 4, 2014, pp. 413–431.

[2] A. K. Ray and B. Chatterjee, "Design of a nearest neighbour classifier System for Bengali character recognition," *IETE Journal of Research*, vol. 30, no. 6, 1984, pp. 226–229.

[3] A. Dutta and S. Chaudhury, "Bengali alpha-numeric character recognition using curvature features," *Pattern Recognition*, vol. 26, no. 12, 1993, pp. 1757–1770.

[4] U. Bhattacharya, M. Shridhar, and S. K. Parui, "On recognition of handwritten Bangla characters," in *Computer Vision, Graphics and Image Processing*, Springer, 2006, pp. 817–828.

[5] T. K. Bhowmik, U. Bhattacharya, and S. K. Parui, "Recognition of Bangla handwritten characters using an MLP classifier based on stroke features, in *International Conference on Neural Information Processing*, Springer, 2004, pp. 814–819.

[6] M. A. Hasnat, S. M. Habib, and M. Khan, "A high performance domain specific OCR for Bangla script," in *Novel Algorithms and Techniques In Telecommunications*, Automation and Industrial Electronics, Springer, 2008, pp. 174–178.

[7] U. Bhattacharya, and B.B. Chaudhuri, "Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals," In *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 3, 2009, pp. 444–457.

[8] M. Biswas, R. Islam, G. K. Shom, M. Shopon, N. Mohammed, S. Momen, et al., "BanglaLekha-Isolated: A multi-purpose comprehensive dataset of handwritten Bangla isolated characters," *Data in Brief*, vol. 12, 2017, pp. 103–107.

[9] S. A. Rabby, S. Haque, M. S. Islam, S. Abujar, and S. A. Hossain, "Ekush: a multipurpose and multitype comprehensive database for online off-Line Bangla handwritten characters," In *Proc. of International Conference on Recent Trends in Image Processing and Pattern Recognition*, 2019, pp. 149–158

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *International Conference on Learning Representations (ICLR)*, 2015.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 770–778.

[12] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, no. 2, Honolulu, HI, USA, 2017, p. 3.

[13] O. Russakovsky, J. Deng, H. Su, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, 2015, pp. 211–252.

[14] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *Proc. International Conference on Learning Representations (ICLR)*, 2015.

International Conference on Bangla Speech and Language Processing (ICBSLP), 27-28 September 2019

Designing a Bangla Stemmer using rule based approach

MD Shahidul Salim Shakib
Computer Science and Engineering department
Khulna University of Engineering & Technology
Khulna,Bangladesh
salim1507034@stud.kuet.ac.bd

Tanim Ahmed
Computer Science and Engineering department
Khulna University of Engineering & Technology
Khulna,Bangladesh
ahmed1507113@stud.kuet.ac.bd

K. M. Azharul Hasan
Computer Science and Engineering department
Khulna University of Engineering & Technology
Khulna,Bangladesh
az@cse.kuet.ac.bd

***Abstract*—Stemming is a preprocessing task for natural language processing that involves normalizing inflected words representing the same concept of the original word. Steaming is a process of text normalization that has many applications. There are many techniques for steaming of inflected words for different languages but very few works for Bangla word steaming. Therefore, stemming Bangla word is a unsolved problem. There are many different situations that can occur in Bangla language for word steaming. In this paper, we present a rule based algorithm to stem Bangla words. We developed the rules for infection detection for verb inflection (বিভক্তি), number inflection (বচন), and others. Using our rules, we developed a system to find the root word of Bangla words and found good performance. Sufficient examples are provided to explain the proposed system.**

Keywords—stemming, text normalization, root word detection, corpus, NLP.

I. INTRODUCTION

Stemming is the process for removing the commoner morphological and inflectional endings from words. The main applications of stemmers are in information retrieval and to increase the recall rate of text mining. There are two categories of stemming namely rule based and corpus based[1]. The rule based stemming removes the suffixes by repeatedly replacing another word(s) or null string. On the other hand, the corpus-based system groups the similar sets that represent the same concept. Popular stemmers are mainly based on rule based system. This is because rule based system is faster and does not require any preprocessing once the rules are generated. The rule based system that encode a large number of language specific rules are available only for a few languages. The most popular and well known stemming approach for English is a rule-based approach namely Porter stemmer [2]. The Porter stemmer successively applies rules to convert a word form into its base form. For example, words such as *hopelessness will be* reduced to *hope* by removing the suffixes *ness* and *less.* There is no effective and full phase Bangla stemmer yet and it is in infant stage[3]. Although there are different kinds of the algorithms for different languages for stemming task, but there is no efficient algorithm for Bangla stemming. In present, there have some works for Bangla word stemming but they follow brute force system. And the accuracy of this system is poor. For the development of Bangla language in Natural Language Understanding(NLU) and Bangla Language Processing, stemming of Bengali words is a must. Due to the morphological structure of Bangla language,it has some difficult issues to design a stemmer. We have proposed a better solution for solving this problem. We developed some important rules to solve this problem. We have divided the words into some categories namely noun, verb, number and others.And proposed an algorithm that will perfectly stem the Bangla words. We also created an exception set to avoid the ambiguity. Our algorithm provides efficiency of 94% and the efficiency can be improved by increasing the terms in the rules. The scheme can be applied to text mining for sentiment analysis[4], Parse design[5], Spam filter[6] and many others.

II. RELATED WORKS

In 1980 “Porter Stemmer” was developed by Martin Porter[2]. Porter stemmer defines some rules these rules applied in a word for removing suffix. Five steps are followed in porter stemmer to stem any word. Several stemming algorithms exist for English such as Lovins Stemming[7], Praice/Husk[8], Dawson[9] and so on. Porter Stemmer is most recognized one that is applied English. Some other stemmers for other languages are Indonesian[10], Malay[11], Dutch[12], Slovene[13], Turkish[14], Latin[15] etc.. There remains some works for stemming Bangla words[16][3][17][1]. Bangla is a language that is highly inflected. It is commonly inflected with noun, verb, adjective. A Light Weight Stemmer for Bengali and its use in spell checker proposed in [17]. [16] describes a clustering-based approach to discover equivalence classes of root words and their morphological variants. Using their clustering technique, they define distance measures to identify equivalent classes. A design a rule-based stemmer for natural language text proposed in [3] using regular expressin syntax. They have another one rule-based stemmer developed for Hindi and Bengali[1] to measure the retrieval effectiveness.

978-1-7281-5242-4/19

III. METHODOLOGY

The proposed stemming approach follows some rules. These rules show how the stemming will be performed. And further an algorithm is developed. This algorithm works based on the rules. The rules are developed with the following general rule.

General rule

Let α is a part of a string of β (α⊂β and α≠β) . Where β is a word and αcan be found at the end of the β. We define the general rule for stemming of the form

α→ v

where α is replaced by v to get the non inflected form of β.If v= ε, then we mean "ε" is an empty string.

Rule for number (বচন) inflection reduction

The general for of the rule is

α→ ε .

For example if β is ছেলেটি then rule is টি→ ε where "টি"is replaced by ε("ε" is an empty string) to get ছেলে. Some possible values of α can be as follows.

টি→ε,[মেয়েটি]ব্যতিক্রমঃখিটিমিটি,বৃষ্টি)

টি→ε,[ছেলেটি]

খানা→ε,[খাতাখানা]

খানি→ε,[বইখানি]

রা →ε,[পাখিরা]

গুলো→ε,[আমগুলো]

গণ →ε,[দেবগণ]

সমূহ→ε,[বৃক্ষসমূহ]

বলি →ε,[পুস্তকাবলি]

গুচ্ছ→ε,[কবিতাগুচ্ছ]

Therefore, we summarize the number (বচন)inflection reduction rules

Rule 1:

বৃন্দ / মন্ডলী / কুঞ্জ / পুঞ্জ / গুচ্ছ / বৃন্দ / সমুদয় / সমূহ / বর্গ / রাশি / আবলি / খানা / গুলি / রাজি / নিকর / খানি / নিচয় / গুলো / মালা / যূথ / সকল / বলি / দের / এরা / দিগ / পাল / দাম / কূল / টা / রা / সব / গণ / টি / ান →ε

Rule for bivokti(বিভক্তি)inflection reduction

The rule for bivokti(বিভক্তি)inflection reduction is α →ε .

For example if β is করাইthen rule is াই→ε where "াই" is replaced by ε to get করাই to কর.Some possible values of α can be as follows.

ায় →ε,[করায়]

াও→ε,[করাও,পড়াও]

াইস→ε , [করাইস]

াচ্ছি →ε,[করাচ্ছি]

াক →ε,[করাক]

াও →ε,[করাও]

াইস→ε,[করাইস]

ালেন→ε,[করালেন]

ালাম→ε,[করালাম]

াইতে→ε,[করাইতে]

াতে→ε,[করাতে](ব্যতিক্রমঃ উৎপাতে)

Therefore, we summarize the rule for bivokti(বিভক্তি) inflection reduction as follows.

Rule 2:

েছ / িয়েছ / ায় / াচ্ছি / ায়েছি / াচ্ছে / াতিস / ালাম / ালেন / াইলে / াইবি / াবেন / াইতে / াতেন / াচ্ছ / াইলি / াতাম / াইবে / াইব / ালি / ালে / াউক / াবো / াইস / ায়ো / াবে / াইও / াইয় / াবি / াতে / াছে / াক / াও / াইয়েছিলাম / াইতেছিলেন / াইয়েছিলাম / াইতেচ্ছিস / াইতেছিস / াইয়েছিলেন / াইয়াছিলেন / াইতেছিলাম / াইয়াছিলে / াইতিছিলি / াইয়েছিলি / াইয়াছিলি / াইয়েছিলে / াচ্ছিলাম / াইতেছিলে / াচ্ছিলেন / াইয়েছিস / াইয়েছিল / াচ্ছিলে / াইয়াছিস / াইতেছেন /াইয়াছেন / াচ্ছিলি / াচ্ছিস / াইয়াছি / াচ্ছেন / াইতেছি / াইয়েছি / াইয়াছে / াইয়েছে / াইতেছে / াইতেছ / াইতেন / াইলেন / াইতাম / াইতিস / াইবেন / াইলাম / াইয়েছ / াইয়াছ / িয়েছেন / িয়েছ / িয়েছিস / িয়েছি / াইলি / েচ্ছিল / িতেছিল / িয়াছিলে / েছিলেন / েছিলে / িয়াছিলি / েছিলি / িতেছিল / ছিল / িয়াছিলেন / িয়াছিলি / েছিলি / িয়াছিলাম / িবে / িবি / িবে / িয়া / িবেন / বেন / িয়া / িস / িয়েছেন / িয়াছি / াইল →ε

Others(বিবিধ)

In this section we present some rules for other inflection detection. The rule is of the form α→ε.For example if β is ‘শুরুতেই’ then rule is তেই→ ε where "তেই" is replaced by εto get শুরুতেইtoশুরু . Some examples are

তে→ε,[ছাড়তে] (ব্যতিক্রমঃহাতে,ভাতে)

ের→ε,[যুগের]

তেই→ε,[শুরুতেই]

তে→ε,[শেয়াকুলিতে]

ের→ε,[গৌরবের]

তে→ε,[দিঘিতে]

টাই→ε,নামটাই]

Therefore we summarize our rule 3 as.

Rule 3:

তে/ ের / তেই / তে / ের / তে/ টাই→ε

Exception

We develop an exception list a part of which is shown in Table 1 that contains the words that has same suffix with the rules we define. These exceptions are not candidate for stemming. This is because if we stem these words then they will lose their real meaning. These words are root words. For example: খোটা,যারা,তারা. Although 'টা' is a suffix in rule 1, but it is not suffix in the word 'খোটা'. Another situation occurs for exception is that '*the whole word*' i.e. (*α=β)* for general rule. For example '**বলি**', '**গুচ্ছ**' can be a single word instead of suffix and this should not be stemmed.

Table 1: Exception list for not to stemming.

Exception
ক্লাস ,দিয়ে, শাসন , পরিচয়, আমাদের, নতুন ,তাদের ,থেকে, এদের, তারা ,যারা, সন্ধান, মার ,পারে ,দিতে, দরকার ,খিটিমিটি, নিয়ে ,খোঁটা

Table 2: Others stemming rules

বিবিধ (Example)		
ঠল → ঠ (উঠল)	রছে →র(করছে)	লের →ল(কমলের)
টাই→ɛ(পাঁচটাই)	জের →জ(তেজের)	েরই →ɛ (পরেই)
ইতেছি-->ɛ(করাইতেছি)	নে →ন(বনে)	রের →র(কবরের)
ও →ɛ(আমারও)	লেম →ɛ(শুনলেম)	টার →ɛ(জলটার)
রলে →র(করলে)	নের →ন(বিসর্জনের)	নীর →নী(ধরনীর)
য়ে →ɛ(পায়ে)	খে →খ(দেখ)	েছি →ɛ(করেছি)
রে →র(করে)	ষ্ঠে →ষ্ঠ(কষ্ঠে)	ড়ে →ড়(পড়ে)
ড়াতে-->ড়(পড়াতে)	লায় →লা(মামলায়)	দের-->ɛ(পরিবারদের)
তেই-->ɛ(শুরুতেই)	ময়ে →ময়(সময়ে)	খি →খ (দেখি)
মার-->মা(প্রতিমার)	ের →ɛ(গৌরবের)	েধেছে→ াধ(বেধেছে)
তে →ত(ভাতে)	লের →ল(সজলের)	তার → তা(রাস্তার)
দ্দর →দ্দ(বরাদ্দর)	লতে →ল(চলতে)	বার→ɛ (বাঁধবার)
	পে →প(মাপে)	

Algorithm1:

*⊲/*An algorithm to remove all possible suffixes from a word*/*

***Input**: st: String of Bangla sentence having n words*

***Output**: A sentence with stemming word*

BanglaStemmer()

```
Begin
Word[i] ←Tokenize (st)/* i=1 to n */
Word[i] ←Remove stop word from the Word[i]
/* i=1 to n */
for i←  1 to n do{
α ← NULL;
     l ← length(word[i])

for j ← ( l-1) down to0 do
α← word[i]
    if(strcmp(α, NotSteamed_suffix)==0) break;
   if(strcmp(α, Bochon_suffix)==0){
    stem[i] = trim(word[i], α);
                // removes α from the end of string
    break;
   }
    if(strcmp(α, Bivokti_suffix)==0){
    stem[i] = trim(word[i], α);
                // removes α from the end of string
    break;
   }
    if(strcmp(α, Other_suffix)==0){
    steam[i] = trim(word[i], α);
                // removes α from the end of string
    break;
   }
} // for j
}// for i
end.
```

After avoiding the exceptions shown in Table 1 we create another rule of the form α → γ, where α is replaced by γ to set the non inflected word. For example পে is replaced by প. Some examples are,

জের→জ,[তেজের]
খি→খ, [দেখি]
ার→া,[বাঁধবার]
নের→ন,[আয়তনের]
তার →তা,[দেবতার]
নের→ন,[বিসর্জনের]
তার → তা,[রাস্তার]
পে → প,[মাপে]
ার → া,[প্রতিমার]

Table 2 summarizes the rule for (বিবিধ) some other stemming rules for called rule 4. Using the rules defined above, we developed an algorithm for Bangla stemmer as shown in Algorithm 1.

IV. RESULTS

For checking accuracy we have taken a part of the famous novel by Rabindronath Thakur "যোগাযোগ" as data set. Table 3 shows the accuracy test using the confusion Matrix. Table 3 shows promising results. The results can be found

in [18].Same as we have taken another article from [19] and Table 3 shows promising results.

.

Table 3: Accuracy Calculation

Data Set	Target to stem	Truly Stemmed	Falsely Stemmed	Not Stemmed	Accuracy
D1	172	162	5	5	94.35%
D2	305	271	26	8	91.47%

V. Conclusion

We developed a rule based algorithm for Bangla stemmer. The approach is general and can be applied to any words. We create rules as well as exception list for word to stemming. The exception list makes the stemmer algorithm to work correctly and to produce good performance avoiding the ambiguity. We found good results in the experimental results. The approach can be made more powerful and accurate by increasing the terms in the rules for substituting *α*. This is a part of an ongoing work. We plan to make a parallel version of the algorithm to improve the response time of the algorithm.

Reference

[1] D. Ganguly, J. Leveling, and G. J. Jones, "Dcu@ fire-2012: rule-based stemmers for bengali and hindi," In: FIRE 2012 Workshop, pp. 17-19, 2012.

[2] M.F. Porter, "*An algorithm for suffix stripping*", Program, 14(3) 1980, pp.130−137.

[3] S. Sarkar and S. Bandyopadhyay, "Design of a rule-based stemmer for natural language text in bengali," in Proceedings of the IJCNLP-08 workshop on NLP for Less Privileged Languages, 2008.

[4] SM Abu Taher, Kazi Afsana Akhter, K M Azharul Hasan " N-Gram Based Sentiment Mining for Bangla Text Using Support Vector Machine", In: International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1-5, 2018.

[5] K M Azharul Hasan, Al-Mahmud, Amit Mondal, Amit Saha, "Recognizing Bangla Grammar using Predictive Parser", International Journal of Computer Science and Information Technology (IJCSIT), 3: 6. pp. 61-73 , 2011.

[6] Karl-Michael Schneider, A comparison of event models for Naive Bayes anti-spam e-mail filtering, Proceeding of EACL '03 Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics, pp. 307-314. 2003.

[7] J.B.Lovins, "Development of a stemming algorithm", *Mechanical Translation and Computational Linguistics 11*, 1968, pp. 22-31.

[8] C.D. Paice, "An evaluation method for stemming algorithms", In the *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, 1990, pp. 42 – 50.

[9] J. Dawson, "Suffix removal and word conflation", *ALLCbulletin, 2(3)*, 1974, pp. 33–46.

[10] V. Berlian, S.N. Vega, and S. Bressan, "Indexing the Indonesian web: Language identification and miscellaneous issues", In the *Tenth International World Wide Web Conference*, Hong Kong. 2001.

[11] S.Y. Tai, C.S. Ong, and N.A. Abdullah, "On designing an automated Malaysian stemmer for the Malay language", (Poster) In the *Proceedings of the fifth international workshop on information retrieval with Asian languages*, Hong Kong, 2000, pp. 207-208.

[12] W. Kraaij and R. Pohlmann, "Viewing stemming as recall enhancement", In the *Proceedings of ACM SIGIR96*, 1996, pp. 40-48.

[13] M. Popovic and P.Willett, "The effectiveness of stemming for natural language access to Slovene textual data", *JASIS, 43 (5)*, 1992, pp.384-390.

[14] F.C. Ekmekcioglu, M.F. Lynch and P.Willett, "Stemming and n-gram matching for term conflation in Turkish texts", *Information Research News, 7 (1)*, 1996, pp. 2-6.

[15] M. Greengrass, A.M. Robertson, S. Robyn, and Willett, "Processing morphological variants in searches of Latin text", *Information research news, 6 (4)*, 1996, pp. 2-5.

[16] P. Majumder, M. Mitra , S. K. Parui,G. Kole, P. Mitra, K. Datta, "YASS: Yet Another Suffix Stripper", ACM Transactions on Information Systems, Vol. 25, No. 4, 2007

[17] M. Islam, M. Uddin, M. Khan, et al., "A light weight stemmer for Bengali and its use in spelling checker," BRAC University, Institutional repository, 2007.

[18]https://github.com/shahidul034/stemmer-accuracy-test

[19]https://www.bd-pratidin.com/first page/2019/09/03/453632

Lyricist Identification using Stylometric Features utilizing BanglaMusicStylo Dataset

Ahmed Al Marouf[1] Rafayet Hossian[2]
Department of Computer Science and Engineering
Human Computer Interaction Lab (HCIRL)
Daffodil International University
Dhaka, Bangladesh
Email: {marouf.cse[1], rafayet3994[2]}@diu.edu.bd

Abstract— **This paper presents a profile-based approach utilizing supervised learning methods to identify the lyricist of Bangla songs written by two legendary poets & novelist Kazi Nazrul Islam and Rabindranath Tagore. The problem statement for this paper could be considered as authorship attribution using stylometric features on Bangla lyrics. We have utilized the BanglaMusicStylo dataset, which consists of 856 and 620 songs of Rabindranath Tagore and Kazi Nazrul Islam, respectively. The traditional authorship attribution works found in the literature are based on the novels written by the authors, not Bangla song lyrics. Using the Bangla song lyrics made it a challenging task, as the word choices made by the authors in songs depends on the rhythms, completeness, situation and many more. In this paper, we have tried to fusion different types of stylometric features, such as lexical, structural, stylistic etc. For experimentation, we have designed the prediction model based on supervised learning exploiting Naïve Bayes (NB), Simple Logistic Regression (SLR), Decision Tree (DT), Support Vector Machine (SVM), and Multilayer Perceptron (MLP). The experimental model consists of several steps including data pre-processing, feature extraction, data processing, and classification model. After performance evaluation, we have got approximately 86.29% accuracy from SLR, which is quite satisfactory.**

Keywords—Authorship Attribution, Linguistic Feature, Stylometric Features, BanglaMusicStylo Dataset, Supervised Learning.

I. INTRODUCTION

Bangla music industry is producing huge number of songs and lyrics of these songs could eventually become an interesting source for natural language processing. The collection of Bangla song lyrics as a dataset is very rare to find in the literature. Formalizing a dataset to apply different natural language processing algorithms such as sentiment analysis, emotional state detection from the song lyrics, genre identification, and authorship attribution could be a wide area of research. With the textual data on the hand, different types of features could be extracted from these special types of texts. Linguistic, psycholinguistic and/or open vocabulary features could be extracted from these textual data and could be utilized to enhance and adopt a machine learning-based approach for finding a solution to the above-mentioned natural language processing problems. We are considering song lyrics as special type of text, as lyrics are written by poets or artists and the written form has different vibes and rhythms. The word choices of the lyricists depends on various components such as rhythms, completeness, situation and many more. Understanding the vocabulary of the lyricist is quite a complicated task. Therefore, it leads to a challenging task to identify the author of the song.

Authorship Attribution [1, 2] is the problem in natural language processing which defined as to find if a given text is authored by a specific writer or not. On the basis of handwritten data [3] or the electronic documents [4] extracting relevant features to distinguish between several authors are performed in literature. The traditional speaker detection from voice inputs is also investigated by G.D. Brown in [5]. Using the independent component analysis method, the traditional cocktail party problem could be solved. Hence, in the context of natural language processing, while compiling the textual data, several types of features are investigated in the literature. Using the Term Frequency-Inverse Document Frequency (TF-IDF) [6] or type-token ratio [7] measurements, authorship identification method had been adopted. In this paper, we have tried to utilize the stylometric features to investigate the efficiency of them for this particular problem using the song lyrics.

BanglaMusicStylo, is a stylometric dataset of Bangla song lyrics of 211 different lyricists [8]. This dataset contains over 2000 Bangla song lyrics having 224,342 words. Based on keyword-based meta searching using the name of the lyricist, title of the songs, genre and emotional words, these song lyrics are digitally archived. As described in [8], this dataset could be utilized for authorship attribution, linguistic forensics, genre classification, vandalism detection, and emotion-based classification. This unique and resourceful dataset is utilized in this paper for identifying different lyricists. Among the 211 Bangla lyricists, we have worked on two most popular poets of Bangla literature Kazi Nazrul Islam and Rabindranath Tagore. This stylometric dataset would be the best possible resource to find the stylometric features from the lyrics. Therefore, determining the stylometric features for this problem and extracting them from lyrics, we find lexical or character-based features, word-based features, stylistic features, and structural features. In this paper, we try to fusion these features and find the most relevant features for each of the lyricists using feature selection or attribute selection algorithms.

In this paper, we have used the textual resources of BanglaMusicStylo dataset and try to model a supervised learning system for identifying lyricists. Another contribution to the knowledge is to comprise the stylometric features, which are used in many application such as email spam detection, emotion detection etc.

978-1-7281-5242-4/19 ©2019 IEEE

Fig. 1. Proposed Methodology for identifying lyricist from Bangla song lyrics.

II. Related Works

In this section, we have discussed the several related works of the problem statement of this paper found in the literature. We have tried to present the contribution of the authors as well as the methods or features adopted for their works.

In [7], M. T. Hossain et al. presented a stylometric analysis on Bengali literature for authorship attribution of six Bangladeshi columnists. They have used the most frequent words, modified word frequency, distribution of some common Bengali words, the spelling of particular words, Bi-grams, Type-Token Ratio (TTR), word & sentence lengths as a feature of the system and cosine similarity as measurement. N-gram and Naïve Bayes based approach has been presented in [9] which has reported around 95% accuracy for three Bangladeshi novelists of novel data namely Humayun Ahmed, Rabindranath Tagore and Shamsur Rahman.

From the nominal and verbal chunks of Bengali language, Hidden Markov Model (HMM) has been applied including the trigram pattern in [10]. They conducted the experiments choosing 15 authors and encoded their literature using UTF-8 encoding training around 50,000 chunks, which reports an accuracy of around 90%. A new corpus containing 3,000 passages authored by three Bengali authors Rabindranath Tagore, Sarat Chandra Chattopadhyay and Bankim Chandra Chattopadhyay are presented in [11]. N-gram based features are extracted and feature selection, the ranking has been reported in their work. N-gram based models are widely used for authorship attribution as they are reported by several researchers in [11-13]. A fusion of n-gram and Naïve Bayes algorithm has shown better accuracy as reported in [9]. A comparative analysis with several methods from literature has been presented by D. M. Anisuzzaman et al. in [9] for novel texts.A graph based approach has been presented in [14] by T. Chakraborty. Describing the limitations of textual features like TF-IDF and TTR, they have adopted and proposed three graph based approaches considering the interaction of character sequences, phraseological patterns and structure of the sentences in the document to construct the graphs. The graph based models are n-gram based character model (NBCM), Chunk based Model (CBM) and Structure based Model (SBM) has reported approximately 95% accuracy. Hence, the adaptability of this model for song lyrics is still a research question.

N. V. G. Raju et al. [15] has proposed an author based rank vector coordinates (ARVC) model. They have utilized the query vector and ARVC for determining the cosine similarity values and the proposed model takes decision from the six different classifiers applied.

The literature shows quite satisfactory works using the novels or Bangla texts, but not for the Bangla song lyrics. To fill up this gap, we have experimented on song lyrics of two legendary poets and lyricists of Bangladesh. Again, for authorship attribution from Bangla lyrics, from our knowledge, there is no work presented in literature. To the best of our knowledge, we are the first to analyze the Bangla lyrics data.

III. Proposed Methodology

This section illustrates the proposed methodology of this paper, as shown in Fig. 1. We have proposed the method using training-testing based approach applying supervised learning classification models. The process takes several steps including data pre-preprocessing, stylometric feature extraction, and classification models to identify the lyricist.

A. Bangla Song Lyrics from BanglaMusicStylo

For our experimentation, we have utilized the BanglaMusicStylo dataset [8]. For our work, we have used only the lyrics of Kazi Nazrul Islam and Rabindranath Tagore. The statistical properties of this dataset is shown in Table I. Therefore, we have worked with total 1476 songs and 96,030 words.

TABLE I. Statistical Properties of BanglaMusicStylo

Properties	Values
No. of Authors	211
Total No. of Songs	2284
Avg. songs per author	13.38389
Avg. words per author	1063.2323
No. of Rabindranath Tagore Songs	856
No. of Words in Rabindranath Tagore Songs	52,784
No. of Kazi Nazrul Islam Songs	620
No. of Words in Kazi Nazrul Islam Songs	43,246

B. Data Pre-processing

Before stepping into feature extraction, we have applied pre-processing on the raw song lyrics. The removal of unnecessary words such as chorus, verse and some digits indicating the repetition of the particular lyrics. As these words does not included in the lyrics, but are written for better understanding of the singer, we have ignored them. The data processing steps are implemented using python NLTK [16] package. The same step is applied on each and every songs of both the lyricists. The less impact of these unnecessary tags are also reported while finding the emotional and language tones [17] and lyrics based title recommendations [18] from English song lyrics.

C. Stylometric Feature Extraction

One of the main contribution of this paper is to analyze the effect of using different categories of stylometric features. For our experimentation, we have extracted several types of stylometric features including word-level features, character-level features, structural features and stylistic features. The total 86 features are classified into four categories, which are listed in Table II.

TABLE II. STYLOMETRIC FEATURES FOR LYRICIST IDENTIFICATION

Feature Number	Description of Feature	Feature Number	Description of Feature
Character-level Features		***Word-level Features***	
F1	Total No. of Character	F63	Total No. of words
F2	Total No. of Letter	F64	No. of words more than 6 character
F3	No. of Whitespace	F65	No. of words with 1 character
F4	No. of Tab space	F66- F77	No. of words with 2 character, 3 character…up to 12 character
F5	No. of Special Character	F78	No. of words more than 12 character
F6	No. of single quote	***Structural Features***	
F7-F12	No. of comma, colon, semicolon, full stop, question mark and exclamation sign	F79	Total no. of line
		F80	Total no. of sentences
		F81	Avg. no. of letter per word
F13-F23	All 11 Shoroborno	F82	Avg. no. of character per word
F24-62	All 39 Banjonborno	***Stylistic Features***	
		F83	No. of short words
		F84	Hapax leomena
		F85	Hapax disleomena
		F86	Yulie's K

The stylometric features are divided into four parts: character-level features, word-level features, structural features and stylistic features. It is to be mentioned that, all these four category of features are determined from each of the song lyrics and are extracted as numeric values.

62 Character-level features are extracted including the unique number of the shorobonrno (11) and banjonborno (39), the punctuation symbols separately. The 16 word-level features are length based as no. of words with 1 character or no. of words with more than 12 characters. These types of unique features are adopted in natural language processing for different purposed. Therefore, it encouraged us to use them as effective features for lyricist identification.

Structural features are depending on the structure of the sentences used in the lyrics, such as total no. of lines in the song, no. of sentences, average no. of letters or character per word. The stylistic features are evidentially used for spam detection, plagiarism detection in literature. Hence, the effect of short words, Hapax leomena, Hapax disleomena and Yulie's K coefficient carries significant impact to the feature extractions, which has been reflected in the experiment analysis. These four category of stylometric features are tested separately for understanding the impact of using these features for authorship attribution, which is the main contribution to the knowledge, of this paper.

D. Classification Model

In this paper, for the experimentation of lyricist identification, we have applied four different traditional classifiers and try to establish a comparative analysis using the classic performance evaluators such as precision, recall, f-measure, accuracy and ROC values. The classifier used are Naïve Bayes (NB) [19], Simple Logistic Regression (SLR) [20], Decision Tree (DT) [21], Support Vector Machine (SVM) [22], and Multilayer Perceptron (MLP) [23]. We have tried to cover different types of classifiers such as probabilistic approach, tree-based, regression-based, vector-based and neural network based classifiers.

IV. EXPERIMENTAL ANALYSIS

For experimentation, we have used Weka [24] tool for applying a supervised learning based model and utilizing the traditional classifiers. As the focus is on the stylometric features, we try to create different scenarios based on the features applied to the model. In Table III, the combinations are illustrated. For each case, we have applied the model in 10-fold cross-validation.

TABLE III. ACCURACY MEASURES OF CLASSIFIERS FOR DIFFERENT FEATURE COMBINATIONS

Feature Combinations	Classifiers				
	NB	**DT**	**SLR**	**SVM**	**MLP**
Character-level features	69.31%	73.73%	82.56%	78.48%	75.08%
Word-level features	60.35%	58.25%	62.67%	60.08%	61.98%
Structural features	46.64%	72.24%	59.81%	58.05%	58.59%
Stylistic features	65.72%	67.28%	69.59%	64.56%	68.64%
All features	72.85%	74.61%	**86.29%**	82.21%	72.84%

Using all the features together, we get the highest accuracy for the proposed system. From Table III, we can see, features separately cannot provide better accuracy, which is achieved by incorporating all the features together. The idea of adding more feature, better accuracy works for this experiment. Another observation is regarding SLR which gives better accuracy (86.29%) than any other classifiers for each of the scenario. The more we have added the number of features, SLR performs better. Table IV demonstrates the performance metrics for all the features through precision, recall, f-measure and Receiver operating characteristic (ROC) value. The Fig. 2 demonstrates the curves for each of the classifiers.

TABLE IV. PERFORMANCE METRICS OF CLASSIFIERS FOR ALL FEATURE COMBINATION

Classifiers	Performance Metrics			
	Precision	**Recall**	**F1-score**	**ROC value**
NB	0.740	0.728	0.714	0.816
DT	0.745	0.746	0.746	0.735
MLP	0.727	0.728	0.723	0.779
SVM	0.830	0.822	0.817	0.801
SLR	**0.864**	**0.863**	**0.861**	**0.928**

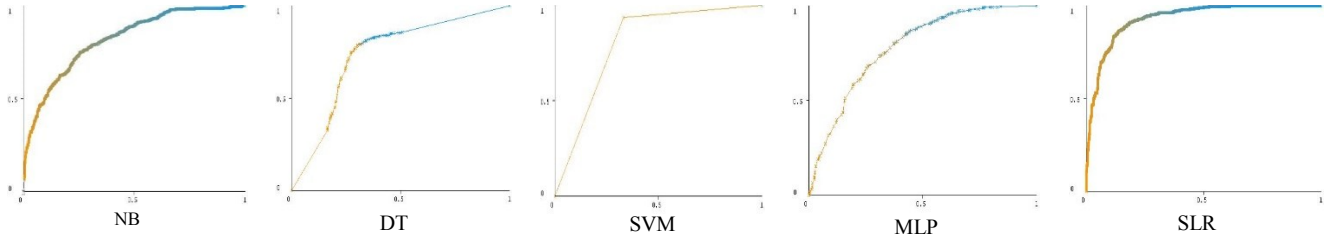


Fig. 2. Receiver operating characteristic (ROC) curves for classifiers.

In our experiment, the problem is defined as two-class classification problem where the classes are of very different sizes, we have measured (using equation 1) the Matthews Correlation Coefficient (MCC) [25] which is more informative than other confusion matrix measures because of balance ratio of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN).

$$MCC = \frac{TP\ X\ TN - FP\ X\ FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad \ldots..(1)$$

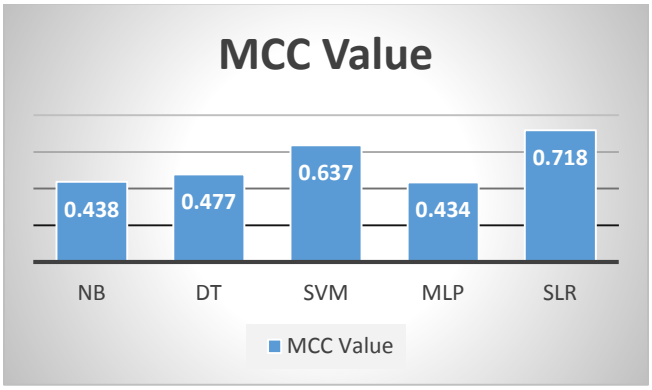


Fig. 3. MCC values for five different classifiers.

V. Conclusion

In this paper, we have presented the feature fusion of stylometric features, which is first time applied on such problem. The proposed process achieved quite satisfactory result for identifying lyricist from song lyrics. The presented supervised learning approach combines four category of features into the system and achieves higher accuracy (86.29% using SLR).

References

[1] Malyutov, M. (2005). Authorship attribution of texts: a review. Electronic Notes in Discrete Mathematics, 21, pp.353-357.

[2] M. Koppel, J. Schler, and S. Argamon, “Computational methods in authorship attribution,” JASIST, vol. 60, no. 1, pp. 9–26, 2009.

[3] G. Knoblich, E. Seigerschmidt, R. Flach and W. Prinz, "Authorship effects in the prediction of handwriting strokes: Evidence for action simulation during action perception", The Quarterly Journal of Experimental Psychology Section A, vol. 55, no. 3, pp. 1027-1046, 2002. Available: 10.1080/02724980143000631.

[4] Agun, H. and Yilmazel, O. (2019). Bucketed common vector scaling for authorship attribution in heterogeneous web collections: A scaling approach for authorship attribution. Journal of Information Science.

[5] G. Brown, S. Yamada and T. Sejnowski, "Independent component analysis at the neural cocktail party", Trends in Neurosciences, vol. 24, no. 1, pp. 54-63, 2001. Available: 10.1016/s0166-2236(00)01683-0.

[6] J. Ramos. Using tf-idf to determine word relevance in document queries. In ICML, 2003.

[7] Hossain, M. T., Rahman, M. M., Ismail, S., & Islam, M. S. (2017). A stylometric analysis on Bengali literature for authorship attribution. 2017 20th International Conference of Computer and Information Technology (ICCIT).

[8] R. Hossain, A. A. Marouf, "BanglaMusicStylo: A Stylometric Dataset of Bangla Music Lyrics", 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), 21-22 Sept. 2018.

[9] D. M. Anisuzzaman, Abdus Salam, "Authorship Attribution for Bengali Language Using the Fusion of N-Gram and Naive Bayes Algorithms", International Journal of Information Technology and Computer Science(IJITCS), Vol.10, No.10, pp.11-21, 2018.

[10] Banerjee, S. "Author Identification in Bengali language." 2013.

[11] Kešelj, Vlado, Fuchun Peng, Nick Cercone, and Calvin Thomas. "N-gram-based author profiles for authorship attribution." In Proceedings of the conference pacific association for computational linguistics, PACLING, vol. 3, pp. 255-264. 2003.

[12] Chakraborty, Tanmoy. "Authorship identification in bengali literature: a comparative analysis." arXiv preprint arXiv:1208.6268 (2012).

[13] Phani, Shanta, ShibamouliLahiri, and Arindam Biswas. "Authorship attribution in bengali language." In Proceedings of the 12th International Conference on Natural Language Processing, pp. 100-105. 2015.

[14] Chakraborty, Tanmoy, and Prasenjit Choudhury. "Authorship identification in Bengali language: A graph based approach." In Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on, pp. 443-446. IEEE, 2016.

[15] Raju, NV Ganapathi, V. Vijay Kumar, and O. Srinivasa Rao. "Author based rank vector coordinates (ARVC) Model for Authorship Attribution." IJIGSP, vol. 8, no. 5 (2016): 68

[16] Bird S., Loper E., “NLTK: The Natural Language Toolkit.”, Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics.

[17] A. A. Marouf, R. Hossain, M. R. K. R. Sarker, B. Pandey and S. M. T. Siddiquee, "Recognizing Language and Emotional Tone from Music Lyrics using IBM Watson Tone Analyzer", 2019 3rd IEEE International conference on Electrical, Computer and Communication Technologies, 20-22, February, 2019.

[18] R. Hossain, M. R. K. R. Sarker, M. Mimo, A. A. Marouf and B. Pandey, "Recommendation Approach of English Songs Title based on Latent Dirichlet Allocation applied on Lyrics", 2019 3rd IEEE International conference on Electrical, Computer and Communication Technologies, 20-22, February, 2019.

[19] I. Rish, “An empirical study of the naive bayes classifier”, In Proceedings of IJCAI-01 workshop on Empirical Methods in AI”, pp. 41–46, Sicily, Italy, 2001.

[20] Niels Landwehr, Mark Hall, Eibe Frank, “Logistic Model Trees”, pp. 161-205, vol. 95, 2005.

[21] S. R. Safavian, D. Landgrebe, “A survey of decision tree classifier methodology”, IEEE Transactions on SMC, pp. 660–674, 1991.

[22] J. C. Platt, "Sequential minimal oprimization: A fast algorithm for training support vector machines", Technical REport MSR-TR_98_14, Microsoft Research, 1998.

[23] S. K. Pal and S. Mitra, “Multilayer perceptron, fuzzy sets, and classification,” IEEE Trans. Neural Networks, vol. 3, pp. 683–697, Sept. 1992.

[24] G. Holmes, A. Donkin and I.H. Witten, “Weka: A machine learning workbench”, Proceedings of Second Australia and New Zealand Conference on Intelligent Information Systems, Brisbane, Australia.

[25] Matthews, B. W. (1975). "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". Biochimica et Biophysica Acta (BBA) - Protein Structure. 405 (2): 442–451.

A Comprehensive Review on Recognition Techniques for Bangla Handwritten Characters

Tapotosh Ghosh
Dept. of Information & Communication Technology
Bangladesh University of Professionals
Dhaka, Bangladesh
16511038@student.bup.edu.bd

Md. Min-ha-zul Abedin
Dept. of Information & Communication Technology
Bangladesh University of Professionals
Dhaka, Bangladesh
16511024@student.bup.edu.bd

Shayer Mahmud Chowdhury
Dept. of Information & Communication Technology
Bangladesh University of Professionals
Dhaka, Bangladesh
mahmudshayer@gmail.com

Mohammad Abu Yousuf
Institute of Information Technology
Jahangirnagar University
Savar, Dhaka-1342, Bangladesh
yousuf@juniv.edu

***Abstract*—Handwritten character recognition is a challenging task in OCR and for a cursive and complex character set like Bangla, it is even harder to implement. Many researchers have proposed different methods for recognizing Bangla Handwritten character set. It is done through analyzing the structure of the characters or through some machine learning process. This paper represents an analysis and overview of the existing methods for recognizing handwritten basic and compound characters. Methods, success rate, limitations and future scope has been mentioned in this paper. The purpose of this paper is to find out the fields in which the systems necessitate improvement and contribute to establish an ideal Bangla Handwritten Character Recognition System.**

Keywords—Handwritten character recognition, Bangla, dataset, accuracy, samples.

I. INTRODUCTION

OCR is a very useful tool in various applications like natural language processing, automatic text entry, automatic reading of characters etc. Bangla characters are complex in shape, very similar and some of them can be written in multiple form. Total number of Bangla basic & compound characters are too many. Handwritten characters also depend on the writer's mood, writing style, quality of pen and paper etc. That is why recognizing Bangla handwritten characters are still considered as one of the biggest challenges for Bangla OCR system.

In the last 2 decades a lot of work has been carried out for Bangla handwritten character recognition. Several image processing, machine learning methods are also proposed & some of them also showed satisfactory result. This paper aims at presenting an analysis on the research procedures of handwritten Bangla character has evolved over decades. We have discussed some of the works related to recognition of handwritten Bangla. We have compared between the existing methods which are related, highlighted the shortcomings and tried to find out some research scopes in this area.

II. THE BANGLA SCRIPT

Bangla has 50 basic letters. Among them, there are 11 vowels & 39 consonants. Bangla is rich in compound characters. There are more than 300 compound letters. They are made up of two or more basic letters. There are 10 modifiers & 10 numeric characters also. Bangla script also used in some other languages such as Meithei and Bhisnupriya Manipuri along with Bengali. Samples of some handwritten Bangla characters are shown in Figure 1.

Figure 1:Some Bangla Handwritten Basic & Compound Character

III. DATABASE

U. Bhattacharya and B. B. Chaudhuri[10] collected 12938 Bangla numerals written by 556 persons. These characters were collected from postal mails, job applications and a set of documents. A data collection form was also created. N.Das[9] et al. created a database of handwritten Bangla compound characters which is available as CMATERdb 3.1.3.3[9]. They collected 55,278 isolated character images, which is of 171 character classes. Biswas et al.[13] created a dataset of 166,105 square images which are of 84 different classes(50 basic, 10 numerals and 32 compound characters).This is known as Banglalekha-isolated[13] database. Rabby A.K.M.S.A. et al.[17] created a database of 367,018 character instances which is known as Ekush[17] database. This database has 122-character classes(50 basic,10 modifiers,10 numerals,52 compound). It is the largest database till now for Bangla characters.

TABLE I: COMPARISON OF DIFFERENT BANGLA CHARACTER DATABASE

Database	No. of classes	Total no. of. Images
U. Bhattacharya et al.[10]	10	12938
CMATERdb 3.1.3.3 [9]	172	55,278
Banglalekha-Isolated [13]	82	1,66,105
Ekush [17]	122	3,67,018

978-1-7281-5242-4/19 ©2019 IEEE

IV. Existing Bangla Handwritten Character Recognition Techniques

Bangla Handwritten Character Recognition (BHCR) system can be broken down into some segments: image acquisition, preprocessing, feature extraction, classification & post processing. Image isolation, noise reduction, labelling are done in image preprocessing. Then the image is given as input to different feature extractor. Later, different classifier such as CNN, MLP, SVM etc. are used to classify the image. A basic architecture of handwritten character recognition system is described in figure 2.

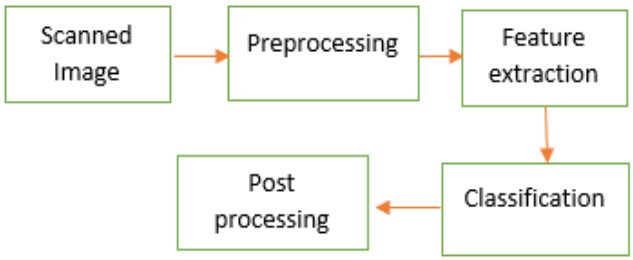


Figure 2:Basic Architecture of a Handwritten Character Recognition System

A. SVM,MLP & ANN Based methods

Researchers have used several methods for feature extraction such as stroke, convex hull, distance based, quad tree based etc. and then classified using MLP, SVM or ANN. Due to change in feature set & technique, they have achieved different recognition accuracy.

Nibaran Das et al (2014) [1] proposed a convex hull-based feature extraction (125 features) method to classify 50 basic characters and 10 numerals. MLP with one hidden layer was chosen for the classification purpose. This method achieved 76.86% & 99.45% accuracy in handwritten basic character & numerals recognition respectively.

Subhadip Basu et al. (2009) [2] proposed a method where characters were extracted from word segmentation. A new feature descriptor (76 features) was also designed.

Nibaran Das et al. (2009) [3] also proposed another feature set containing 132 features were used. Modified shadow features, quad tree based longest run feature, distance-based features, octant and centroid features were included in feature set. Classification was done with MLP in both of the cases. Accuracy improved from 75.05% to 85.40% on 50 basic character classes.

T.K.Bhowmik et al. (2009) [4] proposed a SVM based method. They used RBF, MLP & SVM to classify 45 basic character classes. In this method, image is first classified into a group and then the actual class is identified. This three different two stage hierarchical architecture proved to be more effective than SVM.

Nibaran Das et al (2010) [5] proposed a method where they classified 50 basic and 43 compound characters. They used shadow, quad tree and longest tree feature set. They proposed two methods: one with single layer MLP and other one with SVM classifier. Learning rate was 0.8 and momentum term was 0.7 in MLP classifier. Reported accuracy using MLP is 79.255 and SVM is 81.40%.

U. Bhattacharya et al (2012) [6] proposed a two-stage method where they acquired an accuracy of 95.8% on 50 Bangla basic characters. They used modified quadratic discriminant function (MQDF) to find out the confused pairs and later used a 3-layer MLP with 15 hidden neurons.

K. L. Kabir et al. (2015) [7] proposed a method where different projection-based features such as left projection features, right projection features, quadratic feature algorithm etc. were tested with Quad tree-based features, longest run feature, Octant centroid feature and then classified with ANN. When they used 3 projection-based features set they got the highest accuracy. The accuracy increased as the number of features were increased. Their three-projection based feature set achieved an accuracy of 84.15%.

R. Sarkhel et al. (2015) [8] introduced a region sampling based method. This method selects the regions which contains the most discriminating features of image that describe the character. They have used CMATERdb[9] database and the SVM classifier with RBF kernel. A harmony search-based method was used in this recognition system. They achieved accuracy on Bangla basic characters are 86.533%, for Bangla compound character-set 78.3803% and for mixed character-set 72.8753%.

Nibaran Das et al (2014) [1] proposed a method where only basic characters & numerals were recognized and the acquired accuracy was too low. A different feature set could achieve a better output. Nibaran Das et al (2014) [3] and T.K.Bhowmik et al. (2009) [4] recognized only basic characters, dataset was not rich and accuracy was not satisfactory. Nibaran Das et al (2010) [5] was able to classify 43 compound characters but couldn't achieve good accuracy and database was weak. U. Bhattacharya et al (2012) [6] & K. L. Kabir et al. (2015) [7] proposed recognition methods for 50-60 characters only. Performance of this method in larger dataset and large number of classes is uncertain. R. Sarkhel et al. (2015) recognized a large number of classes but this method did not achieve high accuracy for compound character recognition.

A brief comparison of SVM, MLP and ANN based methods are demonstrated in TABLE I.

TABLE II: Comparison of different SVM,MLP,ANN based Bangla handwritten character Recognition methods

Authors	Method	Used Dataset	classes	Reported accuracy
Nibaran Das et al. (2009) [3]	Feature set (132 features) and MLP	10000 samples	50 basic characters	Test set: 85.40%
T.K. Bhowmik et al. (2009) [4]	RBF, MLP and SVM	27,000 samples	45 basic characters	Test set: 89.22% (highest)
Nibaran Das et al. (2010) [5]	MLP and SVM	19,765 samples	50 basic and 43 compound characters	MLP: 79.25% SVM: 80.51%
U. Bhattacharya et al. (2012) [6]	MQDF and MLP	37,858 samples	50 basic characters	Test set: 95.8%
Nibaran Das et al. (2014) [1]	Convex hull-based feature extraction, MLP	12,000 numerals 10,000 characters	50 basic characters and 10 numerals	Basic: 76.86% Numerals: 99.45%
K. L. Kabir et al. (2015) [7]	Projection based features, ANN	ISI [10]	60 characters	Test set: 84.15%

R. Sarkhel et al. (2015) [8]	Region Sampling, SVM	CMATERdb[9]	231 characters	Basic:86.53%, Compound 78.38% Mixed-72.87%.

B. CNN Based Methods

CNN is one of the most popular method for image recognition. It consists of convolutional layer, pooling layer, fully connected layer and some activation functions. Researchers have proposed several CNN based method for BHCR. Figure 3 demonstrates a basic architecture of CNN.

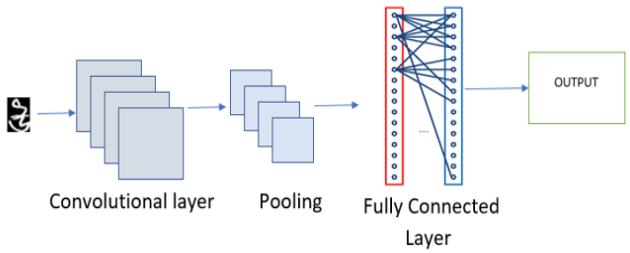


Figure 3: A basic CNN architecture

Md. Mahbubar Rahman et al. (2015) [11] proposed a CNN based method for BHCR.Two convolutional layers with 5×5 kernel, two subsampling layers with 2×2 averaging area , input and output layers were used for classifying 50 handwritten Bangla basic characters. Input layer had 784 neurons and output layer had 50 neurons each for a class. Reported testing accuracy: 85.36%, training accuracy: 94.55%.

M. A. R. Alif et al. (2017) [12] used a modified Resnet architecture to recognize Bangla handwritten characters. They have used dropout layers and Adam optimizer to improve performance of existing Resnet architecture. They have also provided a comparison among other architecture using Banlalekha-Isolated [13] database & CMATERdb[9]. Reported maximum accuracy is 95.18%.

Saikot Roy et al. (2017) [14] proposed a layer wise training deep neural network method for classifying compound Bangla handwritten characters. They also proposed to use RMSProp optimization algorithm which can achieve faster convergence. They achieved 90.33% accuracy in classifying 173 classes of CMATERdb[9].

A. Ashiquzzaman et al. (2017) [15] proposed a CNN method where dropouts were used to reduce overfitting problem and ELU filter was used to reduce gradient vanishing problem to classify 171 compound character classes. They used CMATERdb 3.1.3.3[9] as their database. They achieved 93.68% accuracy in recognizing 171 distinct compound character.

Akm Shahariar Azad Rabby et al. (2018) [16] proposed a CNN architecture which contains 22 layers for recognizing 122 Bangla handwritten character classes. They used Ekushdb[17] for testing and training and CMATERdb[9] for cross validation. Reported accuracy: 97.73% (Ekushdb), 95.01% (CMATERdb).

Akm Shahariar Azad Rabby et al. (2018) [18] proposed a 13 layered CNN with dropout layers. Dropout layers were used to reduce overfitting. Adam optimizer also used in this model. They tested this model on 3 different databases.

Md Zahangir Alom et al. (2018) [19] evaluated performance on different state-of-the-art DCNN model (VGG-net, DenseNet, ALL-Conv, ConvNet, ResNet). They had tested the model on CMATERdb[2] and got finest accuracy using DenseNet.

Sourajit Saha et al. (2018) [20] proposed a DCNN method where divide & conquer and path finder method was used to extract necessary info in every layer. Banglalekha-Isolated [13] database was used. They proposed a method which can manage a variable number of filters without adding load of parameters. They have actually altered GOOGLENET. This method acquired different accuracy varying in different models. The reported highest accuracy is of 97.21%.

A. Fardous et al. (2019) [21] proposed a CNN architecture consisting of 8 convolutional layer, 4 pooling layer, 2 fully connected layer & Relu activation function to classify 171 compound characters. Relu introduced non linearity & dropout reduced overfitting. 95.5% accuracy was reported using this method.

Md. Mahbubar Rahman et al. (2015) [11] and Akm Shahariar Azad Rabby et al. (2018) [18] proposed CNN based architecture which recognized only basic handwritten characters. These methods achieved high accuracy. M. A. R. Alif et al. (2017) [12] and Akm Shahariar Azad Rabby et al. (2018) [18] proposed a system which was validated by more than one database and in all the databases, these methods achieved remarkable accuracy though they classified large number of classes. Md Zahangir Alom et al. (2018) [19] stated a state of art deep CNN architecture where it classified only basic characters and numerals.

A brief comparison of CNN based methods are demonstrated in TABLE II.

TABLE III: COMPARISON OF DIFFERENT CNN BASED BANGLA HANDWRITTEN CHARACTER RECOGNITION METHODS

Authors	Method	Used Dataset	classes	Reported accuracy
Mahbubar Rahman et al. (2015) [11]	CNN	20,000 samples	50 basic characters	Training set: 94.55% Test set: 85.36%
M. A. R. Alif et al. (2017) [12]	Modified Resnet	Banglalekha-Isolated [13] & CMATERdb [9]	84(Banglalekha-Isolated) 231(CMATERdb)	Proposed ResNet18: 95.10% ResNet18: 94.52%, ResNet34: 94.59%
A.Ashiquzzaman et al. (2017) [15]	CNN	CMATERdb [9]	171 compound character classes	Testing set: 93.68%
Saikot Roy et al. (2017) [14]	DCNN	CMATERdb [9]	173 compound characters	Test set: 90.33%
Akm Shahariar Azad Rabby et al. (2018) [16]	CNN	Ekushdb [17] & CMATERdb[9]	122(50 basic,52 compound ,10 numerals, 10 modifiers)	Ekushdb: 97.73% CMATERdb: 95.01%
Akm Shahariar Azad Rabby et al. (2018) [18]	CNN	CMATERdb[9], ISI[10], BanglaLek	50 basic characters	CMATERdb: 98% ISI:96.81% BanglaLekhaIsolated:95.7%

		ha-Isolated[13]		Mixed: 96.40%
Md Zahangir Alom et al. (2018) [19]	DCNN	CMATERdb[9]	73 characters	DenseNet: Alphabet: 98.31% Numerals: 99.13%
Sourajit Saha et al. (2018) [20]	DCNN	Banglalekha-Isolated [13]	84 characters	Test set: 97.21%.
A. Fardous et al. (2019) [21]	CNN	CMATERdb [9]	171 compound character classes	Testing set: 95.5%

C. Other exixsting methods

S. K. Parui et al. (2008) [22] applied a hidden Markov model. They tried a stroke-based recognition technique. A distinct HMM was used for each stroke. 54 groups of strokes were identified and 6 sets of strokes were generated from all the strokes. They worked in 2 stages; First stage: Training with strokes, second stage: Recognition using look up table. Classification accuracy of the proposed system was 87.7% in test case.

K. Roy (2009) [23] introduced an algorithm which could generate a database of strokes based on the database of characters collected for the experiment. A tree-based approach is used for construction of valid character from its constituting strokes. The recognition rate was 96.85% on the test set for the isolated strokes. From the experiment it was obtained 92.92% accuracy for Bangla Character.

M. R. Sazal et al. (2014) [24] implemented a method where deep belief network took raw images as input and processed through unsupervised learning and a supervised fine tuning. They classified 50 basic characters and 10 numerals and obtained 90.27% accuracy after using unsupervised method.

Rahul Pramanik, Soumen Bag (2018) [25] proposed a shape decomposition-based technique where compound characters were segmented into prominent basic shapes in order to achieve better classification and recognition accuracy. They converted the compound character to two or three basic characters to reduce complexity of feature set. At first, letter was decomposed and feature extracted through calculating. Afterwards, they used MLP with 1 hidden layer to classify. They have used ICDAR 2013[10] Segmentation Dataset and CMATERdb [9] for experimental purpose. This method obtained 88.74% accuracy.

K. Roy (2009) [23] and M. R. Sazal et al. (2014) [24] proposed methods only to recognize basic characters and numerals. S. K. Parui et al. (2008) [22] and Rahul Pramanik et al. (2018) [25] proposed techniques to classify some of handwritten compound characters.

All the afore-mentioned techniques could not achieve better accuracy than most of the CNN based techniques. A brief comparison of above-mentioned methods is stated in TABLE III.

TABLE IV: COMPARISON OF DIFFERENT BANGLA HANDWRITTEN CHARACTER RECOGNITION METHODS

Authors	Method	Used Dataset	classes	Reported accuracy
S. K.Parui et al. (2008) [22]	HMM	24500 samples	50 basic characters	Test set: 87.7%
K. Roy (2009) [23]	Stroke based	---	50 characters	Test set: 92.92%
M. M. R. Sazal et al. (2014) [24]	Deep belief network	27900 train & 8600 test samples	50 basic characters and 10 numerals	Basic character: 90.27%
Rahul Pramanik et al. (2018) [25]	Shape decomposition	ICDAR [10], CMATERdb [9]	171 compound classes	Test set: 88.74%.

V. COMPARATIVE STUDY OF DIFFERENT METHODS ACCORDING TO DATASET

A. CMATERdb[9]

TABLE V: COMPARISON OF DIFFERENT HANDWRITTEN CHARACTER RECOGNITION METHODS USING CMATERDB[9] DATABASE

Authors	Method	classes	Reported accuracy
R. Sarkhel et al. (2015) [8]	Region Sampling, SVM	231 characters	Basic:86.53%, Compound 78.38% Mixed:72.87%.
A.Ashiquzzaman et al. (2017) [15]	CNN	171 compound character classes	Testing set: 93.68%
Saikot Roy et al. (2017) [14]	DCNN	173 compound characters	Testing set: 90.33%
Akm Shahariar Azad Rabby et al. (2018) [16]	CNN	122(50 basic,52 compound, 10 numerals, 10 modifiers)	Testing set: 95.01%
Akm Shahariar Azad Rabby et al. (2018) [18]	CNN	50 basic characters	Testing set: 98%
Md Zahangir Alom et al. (2018) [19]	DCNN	73 characters	DenseNet: Alphabet: 98.31% Numerals: 99.13%
A. Fardous et al. (2019) [21]	CNN	171 compound character classes	Testing set: 95.5%
Rahul Pramanik et al. (2018) [25]	Shape decomposition	171 compound classes	Test set: 88.74%.

B. Banglalekha-Isolated[13]

TABLE VI: COMPARISON OF DIFFERENT HANDWRITTEN CHARACTER RECOGNITION METHODS USING BANGLALEKHA-ISOLATED[13] DATABASE

Authors	Method	classes	Reported accuracy
M. A. R. Alif et al. (2017) [12]	Modified Resnet	84(Banglalekha-Isolated)	Proposed ResNet18: 95.10% ResNet18: 94.52%, ResNet34: 94.59%
Akm Shahariar Azad Rabby et al. (2018) [18]	CNN	50 basic characters	BanglaLekhaIsolated: 95.7%
Sourajit Saha et al. (2018) [20]	DCNN	84 characters	Test set: 97.21%.

C. ISI[10]

TABLE VII: COMPARISON OF DIFFERENT HANDWRITTEN CHARACTER RECOGNITION METHODS USING ISI[10] DATABASE

Authors	Method	classes	Reported accuracy
K. L. Kabir et al. (2015) [7]	Projection based features, ANN	60 characters	Test set: 84.15%
Akm Shahariar Azad Rabby et al. (2018) [16]	CNN	122(50 basic,52 compound, 10numerals, 10 modifiers)	Testing set: 95.01%
Rahul Pramanik et al. (2018) [25]	Shape decomposition	171 compound classes	Test set: 88.74%.

VI. CHALLENGES FOR BANGLA HANDWRITTEN CHARACTER RECOGNITION

- Bangla character set is vast consisting nearly 400 characters in total. Designing a recognition system for such huge number of class is very challenging.
- Some of the characters are very similar in shape and structure.
- Size of Bangla characters are not uniform.
- Each character might be written in different structure by different person.

VII. FUTURE SCOPE

Most of the researchers have worked with basic and frequently used compound characters. No established method has been found where all the basic & compound characters (total around 400 characters) are recognized with good accuracy. Unsupervised learning techniques has not been yet used in this area. Hence, there is a scope for using this type of techniques in BHCR.

VIII. CONCLUSION

Voluminous works on BHCR have been carried out so far. In this paper, we have tried to throw a light on various research works which have been executed and achieved great accuracy. Review of some of the methods in details showing comparison between different works and methods of BHCR have also been accomplished in this paper. Challenges faced in recognition of Bangla handwritten character have been assessed here. It is also observed that CNN based methods gained more accuracy than other applied procedures, in addition to this, altering layers and filters can achieve more accuracy. However, no established method has yet been discovered which can recognize all the basic and compound characters efficiently.

REFERENCES

[1] Das, Nibaran & Pramanik, Sandip & Basu, Subhadip & Saha, Punam & Sarkar, Ram & Kundu, Mahantapas & Nasipuri, Mita. (2014). Recognition of Handwritten Bangla Basic Characters and Digits using Convex Hull based Feature Set. 10.13140/2.1.3689.4089.

[2] Basu, Subhadip & Das, Nibaran & Sarkar, Ram & Kundu, Mahantapas & Nasipuri, Mita & Kumar Basu, Dipak. (2009). A hierarchical approach to recognition of handwritten Bangla characters. Pattern Recognition. 42. 1467-1484. 10.1016/j.patcog.2009.01.008.

[3] Das, Nibaran & Basu, Subhadip & Sarkar, Ram & Kundu, Mahantapas & Nasipuri, Mita & kumar Basu, Dipak. (2009). An Improved Feature Descriptor for Recognition of Handwritten Bangla Alphabet. Proc. of International conference on Signal and Image Processing (ICSIP-2009), India.

[4] Bhowmik, T.K., Ghanty, P., Roy, A. et al. IJDAR (2009) 12: 97. https://doi.org/10.1007/s10032-009-0084-x

[5] Das, Nibaran & Das, Bindaban & Sarkar, Ram & Basu, Subhadip & Kundu, Mahantapas & Nasipuri, Mita. (2010). Handwritten Bangla Basic and Compound character recognition using MLP and SVM classifier. Journal of Computing. 2.

[6] Bhattacharya, U., Shridhar, M., Parui, S.K. et al. Pattern Anal Applic (2012) 15: 445. https://doi.org/10.1007/s10044-012-0278-6

[7] K. L. Kabir et al., "Projection-based features: A superior domain for handwritten Bangla basic characters recognition," 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, 2015, pp. 1-7.

[8] R. Sarkhel, A. K. Saha and N. Das, "An enhanced harmony search method for Bangla handwritten character recognition using region sampling," 2015 IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS), Kolkata, 2015, pp. 325-330. doi: 10.1109/ReTIS.2015.7232899

[9] N. Das, K. Acharya, R. Sarkar, S. Basu, M. Kundu, M. Nasipuri, A benchmark image database of isolated Bangla handwritten compound characters, Int. J. Docum. Anal. Recogn. 17 (4) (2014) 413–431.

[10] U. Bhattacharya and B. B. Chaudhuri, "Databases for research on recognition of handwritten characters of Indian scripts", In Proc. of the 8th Int. Conf. on Document Analysis and Recognition (ICDAR-2005), Seoul, Korea, vol. II, page: 789-793, 2005.

[11] Mahbubar Rahman, Md & Akhand, M. A. H. & Islam, Shahidul & Chandra Shill, Pintu & Rahman, M. M.. (2015). Bangla Handwritten Character Recognition using Convolutional Neural Network. International Journal of Image, Graphics and Signal Processing (IJIGSP). 7. 42-49. 10.5815/ijigsp.2015.08.05.

[12] M. A. R. Alif, S. Ahmed and M. A. Hasan, "Isolated Bangla handwritten character recognition with convolutional neural network," 2017 20th International Conference of Computer and Information Technology (ICCIT), Dhaka, 2017, pp. 1-6.

[13] Biswas, Mithun & Islam, Rafiqul & Kumar Shom, Gautam & Shopon, Md & Mohammed, Nabeel & Momen, Sifat & Abedin, Anowarul. (2017). BanglaLekha-Isolated: A multi-purpose comprehensive dataset of Handwritten Bangla Isolated characters. Data in Brief. 12. 103-107. 10.1016/j.dib.2017.03.035.

[14] Roy, Saikat & Das, Nibaran & Kundu, Mahantapas & Nasipuri, Mita. (2017). Handwritten Isolated Bangla Compound Character Recognition: a new benchmark using a novel deep learning approach. Pattern Recognition Letters. 90. 10.1016/j.patrec.2017.03.004.

[15] A. Ashiquzzaman, A. K. Tushar, S. Dutta and F. Mohsin, "An efficient method for improving classification accuracy of handwritten Bangla compound characters using DCNN with dropout and ELU," 2017 Third International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN), Kolkata, 2017, pp. 147-152.

[16] Rabby, Akm Shahariar Azad & Haque, Sadeka & Abujar, Sheikh & Hossain, Syed. (2018). EkushNet: Using Convolutional Neural Network for Bangla Handwritten Recognition. Procedia Computer Science. 143. 603-610. 10.1016/j.procs.2018.10.437.

[17] Rabby A.K.M.S.A., Haque S., Islam M.S., Abujar S., Hossain S.A. (2019) Ekush: A Multipurpose and Multitype Comprehensive Database for Online Off-Line Bangla Handwritten Characters. In: Santosh K., Hegadi R. (eds) Recent Trends in Image Processing and Pattern Recognition. RTIP2R 2018. Communications in Computer and Information Science, vol 1037. Springer, Singapore.

[18] Rabby, Akm Shahariar Azad & Haque, Sadeka & Islam, Sanzidul & Abujar, Sheikh & Hossain, Syed. (2018). BornoNet: Bangla Handwritten Characters Recognition Using Convolutional Neural Network. Procedia Computer Science. 143. 528-535. 10.1016/j.procs.2018.10.426.

[19] Md Zahangir Alom, Paheding Sidike, Mahmudul Hasan, Tarek M. Taha, and Vijayan K. Asari, "Handwritten Bangla Character Recognition Using the State-of-the-Art Deep Convolutional Neural Networks," Computational Intelligence and Neuroscience, vol. 2018, Article ID 6747098, 13 pages, 2018.

[20] Saha, Sourajit & Saha, Nisha. (2018). A Lightning fast approach to classify Bangla Handwritten Characters and Numerals using newly structured Deep Neural Network. Procedia Computer Science. 132. 1760-1770. 10.1016/j.procs.2018.05.151.

[21] A. Fardous and S. Afroge, "Handwritten Isolated Bangla Compound Character Recognition," 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE), Cox'sBazar, Bangladesh, 2019, pp. 1-5.

[22] S. K. Parui, K. Guin, U. Bhattacharya and B. B. Chaudhuri, "Online handwritten Bangla character recognition using HMM," 2008 19th International Conference on Pattern Recognition, Tampa, FL, 2008, pp. 1-4.

[23] Roy, Kaushik. (2009). Stroke-Database Design for Online Handwriting Recognition in Bangla. International Journal of Modern Engineering Research..

[24] M. M. R. Sazal, S. K. Biswas, M. F. Amin and K. Murase, "Bangla handwritten character recognition using deep belief network," *2013 International Conference on Electrical Information and Communication Technology (EICT)*, Khulna, 2014, pp. 1-5.

[25] Rahul Pramanik, Soumen Bag, Shape decomposition-based handwritten compound character recognition for Bangla OCR, Journal of Visual Communication and Image Representation, Volume 50,2018,Pages 123-134.

Analyzing Sentiment of Movie Reviews in Bangla by Applying Machine Learning Techniques

Rumman Rashid Chowdhury*, Mohammad Shahadat Hossain†, Sazzad Hossain‡ and Karl Andersson§
*Department of Computer Science and Engineering, University of Chittagong, Chittagong, Bangladesh
Email: rumman179@gmail.com
†Department of Computer Science and Engineering, University of Chittagong, Chittagong, Bangladesh
Email: hossain_ms@cu.ac.bd
‡Department of Computer Science and Engineering, University of Liberal Arts Bangladesh, Dhaka, Bangladesh
Email: sazzad.hossain@ulab.edu.bd
§Department of Computer Science, Electrical and Space Engineering, Luleå University of Technology Skellefteå, Sweden
Email: karl.andersson@ltu.se

***Abstract*—This paper proposes a process of sentiment analysis of movie reviews written in Bangla language. This process can automate the analysis of audience's reaction towards a specific movie or TV show. With more and more people expressing their opinions openly in the social networking sites, analyzing the sentiment of comments made about a specific movie can indicate how well the movie is being accepted by the general public. The dataset used in this experiment was collected and labeled manually from publicly available comments and posts from social media websites. Using Support Vector Machine algorithm, this model achieves 88.90% accuracy on the test set and by using Long Short Term Memory network [1] the model manages to achieve 82.42% accuracy. Furthermore, a comparison with some other machine learning approaches is presented in this paper.**

Keywords — Bangla sentiment analysis, Support Vector Machines, Long Short Term Memory.

I. Introduction

In this Indian subcontinent, Bangla is the language with the second-highest number of speakers and it holds the sixth position among the most spoken languages of the world. During the last decade, with the increase of the use of social media, people are expressing their opinions on various topics using the Facebook, Twitter etc social networking websites, often in their own native language. The use of Bangla language on social media has also escalated since many easy to use Bangla keyboard apps were introduced in the last few years. People often discuss about movies and TV shows on the social networking sites. There are even dedicated groups for people just to discuss topics related to these. By analyzing the sentiment of the comments made by people towards a specific movie or TV show, it is possible to know if people are considering the movie positively or not. Another practical use case might be to analyze the reaction of the audience towards the trailer of a movie, which can indicate whether the movie is anticipated by the general public positively or negatively. But analyzing every single comment manually is a long and tedious task. Therefore, this paper discusses the performance of some machine learning models for analyzing the sentiment of movie related comments made in Bangla language. Various machine learning methods were applied, such as Support Vector Machine [2] and Multinomial Naive Bayes on this dataset. As Deep Learning based approaches are being used in various sectors recently [3] [4], Long Short Term Memory [1] (which is an improved version of Recurrent Neural Network) was also applied for comparison. By providing a method for automated sentiment analysis, this research opens up the path for further development of sentiment analysis [5] methods of Bangla language in other sectors too.

The remainder of this article is structured as follows: Section II covers related work on Bangla sentiment analysis, Section III briefly discusses about the dataset and preprocessing techniques used in this experiment while Section IV provides an overview of the methodology and system architecture. Section V describes the experiment process and presents the results and Section VI concludes the paper describing the future scope of this research.

II. Related Work

Although a lot of research has been done on sentiment analysis of English movie reviews using the IMDb (Internet Movie Database) dataset, there has not been much research performed on movie reviews in Bangla, mostly due to the lack of sufficient data. The paper entitled "Evaluation of Naive Bayes and Support Vector Machines on Bangla Textual Movie Reviews" by Hafizur Rahman et al. [6] in 2018 compares the performance of Naive Bayes and SVM in classifying movie reviews in Bangla. The dataset contains 800 comments. It was collected by the authors by using web crawling methods from Bangla movie review sites and social media. In this paper, the performance of the models were judged by the recall and

978-1-7281-5242-4/19

precision values. In their experiment, SVM provided the best precision and recall of 0.86. N-gram Based Sentiment Mining for Bangla Text Using Support Vector Machine by Taher et al. [7] approaches the sentiment analysis problem primarily using SVM for classification and N-gram method for vectorization. An interesting technique used in this paper was Negativity Seperation, which separates the negative postfix of a word from the actual word, thus putting more emphasis on the fact that the overall sentence contains negativity. Furthermore, a comparison between Linear and Non-linear SVM was presented, which indicates that Linear SVM performed better in case of text classification. An experiment on Detecting Multilabel Sentiment and Emotions from Bangla YouTube Comments was preformed by Irtiza et al. [8] where their Deep Learning based [9] LSTM approach achieves 65.97% accuracy in a three label dataset and 54.24% accuracy in a five label dataset. Mahtab et al. [10] presented a research work of Sentiment Analysis on Bangladesh Cricket with Support Vector Machine. The dataset named ABSA that was used here contains 2979 data samples. The data samples were labeled as three classes, namely Positive, Negative and Neutral. The authors also collected a dataset of their own and it contains 1601 data samples with three classes. Python NLTK was used for tokenizing and TF-IDF Vectorizer for vectorization. Accuracy on the custom dataset was 64.596% and 73.49% on the ABSA dataset.

From the above discussion it can be interpreted that, the main reason that the Bangla Natural Language Processing sector is being held back, is the lack of sufficient data. A previous research on movie review only managed to collect 800 data samples. The dataset that was built for our experiment contains around 4000 data samples, thus the amount of data contributes in improving the performance of the machine learning models. Furthermore, Deep Learning models require even more data as they perform feature extraction automatically based on training data samples [11]. Therefore, classification using deep learning is difficult with the quantity of data currently available. This paper focuses on binary classification (positive and negative) of the documents, thus being relatively simpler for the deep learning classifiers.

III. Dataset Preprocessing and Document Representation

The dataset used in this experiment was collected manually from the comments of people in social networking websites. It contains around 4000 samples, each labeled as either positive or negative. Due to the less amount of data, further classification of the positive and negative classes was not possible using this particular dataset. 80% of the data was used for training and the remaining 20% was used as the test set. For validating the performance even further, K-fold cross validation was performed. Some samples from the dataset are as follows:

neg কত একশন মুভি দেখলাম ,কিন্তু কোনোটাই ভাল লাগে নি।

neg আমার দেখা এই বছর এর সবচেয়ে খারাপ ফিল্ম বলতে হবে এই মুভিটিকে।

neg পরের সিজনগুলা কিছু কিছু এপিসোড বাদে লেম মনে হয়ছে।

pos শেষ কবে কোন মুভি দেখে আমার চোখে পানি এসেছে, আমার মনে নেই। তবে এই মুভি দেখা শেষে চোখের কোণে পানি ছিলো আমার।

pos হাজারো বাজে সিনেমার ভিড়ে একটি মানসম্মত সিনেমা।

pos অসাধারণ, ভয়েস এক্টিং মুগ্ধ হওয়ার মত এবং ক্যারেক্টার ডেভেলপমেন্ট খুব সুন্দরভাবে ফুটিয়ে তোলা হয়েছে।

A. Preprocessing

The raw data that was collected is not suitable for classification directly. It contains a lot of punctuation marks, emojis etc which are irrelevant for the sentiment analysis process. The dataset needs to be preprocessed before starting the classification process for getting better accuracy. There are many pre-processing steps that are widely applied on the datasets depending on the language of the dataset. Preprocessing is a vital step before the initiation of classification process. The result of classification depends on successful preprocessing steps. Some mentionable methods that are used are tokenization, punctuation and emoticon removal, stemming, stop-word removal etc.

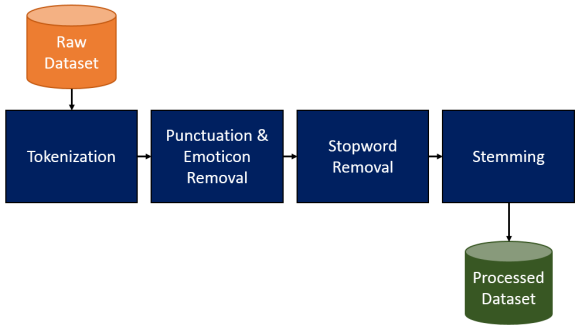


Figure 1. Preprocessing workflow

1) Tokenization and Punctuation Removal: Tokenization means breaking up a given text into meaningful units called tokens. The tokens may be words, numbers or punctuation marks. In this experiment the words were separated by splitting a sentence based on the spaces. While performing tokenization on each data sample, the unnecessary items in the data, such as punctuation marks, alphabets of other languages, emoticons etc. were also removed. After this step, an array containing sub-arrays of tokenized data was generated. A seperate array was declared to store the labels.

2) Stopword Removal: The words whose importance in the text corpus is negligible are known as stopwords. These words have no significance while classifying documents. In case of English, words like ”a”, ”of”, ”the”, ”for”, ”my” etc. are stopwords. Similarly in Bangla, the words

"অতএব", "অথচ", "অথবা", "অনুযায়ী", "এটা", "এটাই", "এটি" etc. are considered as stopwords. The list of stopwords in Bangla was collected from [12]. For example, from the sentence "ছোট এপিসোড হলেও অনেক মজাদার", the following tokens are obtained after stopword removal: [ছোট, এপিসোড, হলেও, মজাদার]. Here "অনেক" was considered as a stopword, therefore it was excluded.

3) Stemming: The term stemming specifies reducing variations of a word into its basic form. There can be different forms of a word based on the context it is being used. For instance, "করা", "করছি", "করছিলাম", "করছিলে", "করেছে", "করেছি" etc. for all these words, "কর" is the root word. The main purpose of stemming is to reduce conjugational forms of a word to a common basic form. In this way the total number of words that the classifier has to work with, can be decreased by a huge margin. For performing this procedure, the common prefix and postfixes that are used in Bangla words were stored in an array. Python Regular Expression library was utilized to detect the prefix and postfixes in the words and the trimmed version of the words were added to the new processed corpus. In this sentence "ছোট এপিসোড হলেও অনেক মজাদার", after stemming the words become as follows (excluding stopwords): [ছোট, এপিসোড, হল, মজা] . Here, "হলেও" is changed into it's base form "হল".

B. Document Representation

Document representation is a preprocessing technique that can reduce the complexity of a dataset and make it easier for the machine learning model to handle. The document has to be transformed from the current text version to a vector representation. One of the most commonly used document vector representation is the vector space model [13], where documents are represented by vectors of words. For vector representation and feature extraction, Tf-Idf Vectorizer and Count Vectorizer were used in this experiment.

1) Count Vectorizer: The CountVectorizer creates a vocabulary of the words in the corpus and counts the frequencies of the words. It is also used to encode new text documents using the generated vocabulary.

2) Tf-Idf Vectorizer: TF-IDF refers to term frequency-inverse document frequency. It is a statistical metric for evaluating the importance of a word in a text document in a corpus. The importance increases in proportion to the frequency a word appears in the document and decreases when the word occurs in the corpus more frequently. The TfidfVectorizer function from scikit-learn creates a matrix of TF-IDF features from a collection of raw documents.

IV. METHODOLOGY AND SYSTEM ARCHITECTURE

A wide variety of algorithms can be applied for the task of sentiment analysis. Most of the time the performance of each method varies depending on the dataset. During this research, classic machine learning algorithms such as, Support Vector Machine (using Count Vectorizer and Tf-Idf Vectorizer) and also deep learning based [14] methods like Long Short Term Memory network were implemented. The workflow of the model construction process is illustrated in figure 2.

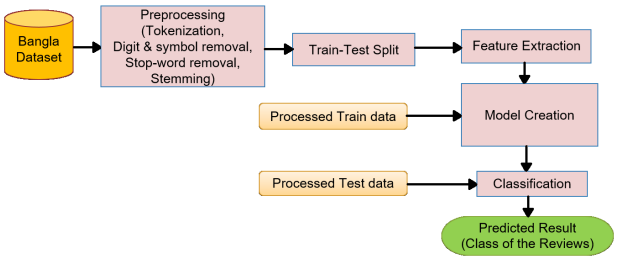


Figure 2. Model Construction workflow

A. Support Vector Machine

Support vector machine algorithms aim to find a hyperplane in an N-dimesional space that can distinctly classify the data points. For separating two classes of data samples, there can be a lot of possible hyperplanes to be chosen from, but the objective of the Support Vector Machine algorithm is to find the one which has the maximum margin distance between the data points of the two classes. By the maximization of the margin the model can classify future data samples more accurately. The hyperplane is a decision boundary that assists in the classification of the data points. Support vectors are the points that are close to the hyperplane and the margin of the hyperplane is calculated based on the support vectors.

In this experiment, the LinearSVC (Linear Support Vector Classifier) function from the scikit-learn library was used. First, the data was vectorized using the CountVectorizer function from scikit-learn. The minimum document frequency parameter was set to 2, so only the words which occurred more than once will be considered. The parameter for Ngram range was set to (1,3) so, the vocabulary will be created including 1,2 and 3-gram sequences. For example, from "দারুন হয়েছে মুভিটা" we can get the following volabulary ["দারুন", "হয়েছে", "মুভিটা", "দারুন হয়েছে", "হয়েছে মুভিটা", "দারুন হয়েছে মুভিটা"]. By using n-grams form training the model, the model will be able to learn more complex features within the text data, like the co-occurring sequences of words that can have a specific meaning. After vectorization, the training data is fit to the model by using the LinearSVC function. 20% of the data is kept unseen from the model, so that they can be used to evaluate the model's classification capabilities. After the training session is complete, the model is then used to classify the test set.

Similarly, the experiment was performed with Tf-Idf Vectorizer function from scikit-learn library. The minimum document frequency parameter was set to 2 and the

parameter for Ngram range was set to (1,3). 20% of the data was used as test set. After performing vectorization using TfidfVectorizer function, the training data is fit to the model by using the LinearSVC function.

To gain further insight about the capability of the model, K-fold cross validation was also applied, with 5 folds in the dataset.

B. Long Short Term Memory

Long Short Term Memory network is an improved version of Recurrent Neural Network, which solves the gradient vanishing issue that occurs in Recurrent Neural Networks. Neural Networks normally classify each input individually, without any context. The previous data samples have no influence on the current data sample. But in case of sentences, the series of words can often mean something which might not be interpretable when the individual words are taken into account. This is why Recurrent Neural Networks are being widely used in the Natural Language Processing sector recently, as they can preserve the context by considering the previous inputs while classifying. The preprocessing part was a little different than that of SVM in this case. After reading and cleaning the corpus, removing stopwords and stemming the words, the sentences are passed to a tokenizer which creates a vocabulary of the words in the sentences and creates integer arrays for each sentence by replacing each word with their integer value in the vocabulary. The maximum length of the sentences is set to 20 words, as most comments are shorter than that. The longer sequences are truncated and the shorter ones are padded with zeroes on the left. The LSTM model used in this experiment was implemented using Tensorflow [15] and Keras [16]. It consists of an Embedding layer with input dimension equal to the vocabulary size and output dimension equal to 256. Input length is equal to 20, which is the maximum length of each sequence. The Embedding layer is followed by an LSTM layer with 128 units. There is a Fully connected layer with 64 nodes after the LSTM layer whose activation function is ReLU (Rectified Linear Unit) [17]. ReLU is used for introducing non-linearity in the output of the nodes. Finally the output layer consists of 2 nodes, equal to the number of classes. The activation function of the output layer was set as 'softmax'. Softmax provides probabilistic values for each class [18]. The classifier was compiled with optimization function set as 'Adam' [19] and loss function as 'categorical_crossentropy' [20].

V. Experiment and Result Analysis

Using SVM with Count Vectorizer, the model managed to achieve 87.016% accuracy on 20% of the dataset as the test set. With Tf-Idf Vectorizer, the model achieved 88.90% accuracy. A confusion matrix of the prediction results is displayed in figure 3. The details about the model's performance is portrayed in table I.

Table I
Performance metrics of the SVM model

Vectorizer	Accuracy	Recall	Precision	F1 Score
Count	87.02%	0.88	0.86	0.87
Tf-Idf	88.90%	0.89	0.88	0.89

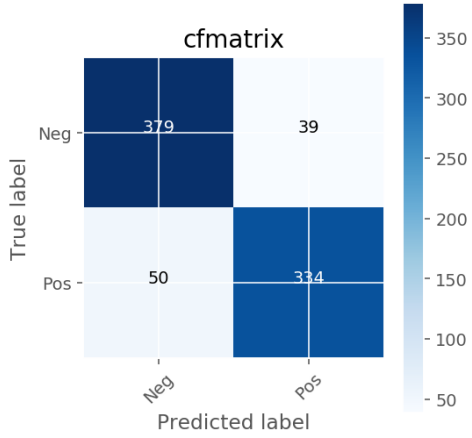


Figure 3. Confusion Matrix of the predictions made by SVM (Tf-Idf) model

For further validation of the model, KFold Cross Validation was applied with 5 folds. The accuracy of SVM with Tf-Idf Vectorizer in the five iterations were as follows: 80.92%, 80.04%, 74.91%, 78.40%, 88.64%, with an average of 80.58% accuracy. The accuracy of SVM with Count Vectorizer in the five iterations were as follows: 79.80%, 79.18%, 76.03%, 76.90%, 87.02%, with an average of 79.786% accuracy.

The experiment was also performed using the Multinomial Naive Bayes algorithm to classify the sentiment of the data samples. This model managed to obtain 88.38% accuracy on the 20% test data. With K-Fold cross validation and number of splits set to 5, the average accuracy was 79.36%.

Using LSTM the model was trained for 12 epochs. Each epoch took approximately 6 seconds to complete. Although the number of epochs was set to 50, the Keras EarlyStopping callback stopped the training at 12 epochs to prevent overfitting as the validation accuracy was not improving anymore. The LSTM model with the best accuracy was saved using the ModelCheckpoint callback and it managed to achieve a maximum of 82.42% validation accuracy on the test data. The f1-score obtained by the best model was 0.83. The validation accuracy graph of the training phase is illustrated in figure 4 and a confusion matrix of the prediction results is displayed in figure 5.

VI. Conclusion and Future Work

This research explores the different methods that can be used to perform sentiment analysis on Bangla movie

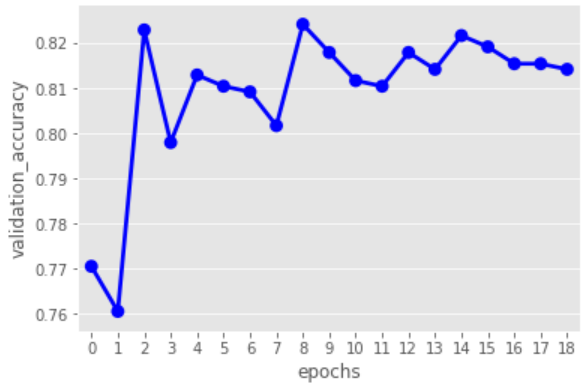


Figure 4. Validation accuracy graph of the LSTM training session

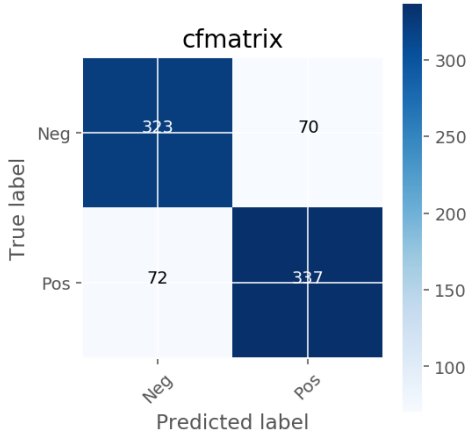


Figure 5. Confusion Matrix of the predictions made by LSTM model

review samples. By analyzing the results, it can be interpreted that SVM based models performed better than other approaches. This model also obtained decent accuracy when K-fold cross validation was applied, which displays the versatility of the system. As the dataset used in this experiment was relatively small and less complex and linearly separable, the traditional Machine Learning techniques performed well. However, the results may vary depending on the complexity and quantity of data samples, as deep learning models can extract complex features from the data much more effectively. The models that were generated in this experiment can be utilized by the film producers to conveniently analyze people's comments from the social media about their movie, as social networking websites reflect the views of people in a very clear manner.

A comparison between the performance of the approaches is portrayed in table II.

For further development of this research, the dataset needs to be enhanced by adding more data samples. It will be interesting to see how the model performs while working with a large dataset like the IMDb review dataset in

Table II
Performance comparison between the models

Classifier	Accuracy
SVM (Tf-Idf)	88.90%
SVM (Count)	87.02%
Multinomial Naive Bayes	88.38%
Long Short Term Memory	82.42%

Bangla. Although currently the classic Machine Learning algorithms are performing well, the Deep Learning based models have the potential to exceed their current levels of performance if adequate amount of data can be provided. Some other extensions are also possible. For example, by integrating knowledge driven methodology such as Belief Rule Base (BRB), which is widely used where uncertainty becomes an issue [21] [22] [23] [24] [25].

References

[1] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[2] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of machine learning research*, vol. 2, no. Nov, pp. 45–66, 2001.

[3] R. Chowdhury, M. Hossain, R. Islam, K. Andersson, and S. Hossain, "Bangla handwritten character recognition using convolutional neural network with data augmentation," 04 2019.

[4] T. Ahmed, S. Hossain, M. Hossain, R. Islam, and K. Andersson, "Facial expression recognition using convolutional neural network with data augmentation," 04 2019.

[5] S. Chowdhury and W. Chowdhury, "Performing sentiment analysis in bangla microblog posts," in *2014 International Conference on Informatics, Electronics & Vision (ICIEV)*. IEEE, 2014, pp. 1–6.

[6] N. Banik and M. Hasan Hafizur Rahman, "Evaluation of naïve bayes and support vector machines on bangla textual movie reviews," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–6.

[7] S. Abu Taher, K. Afsana Akhter, and K. M. Azharul Hasan, "N-gram based sentiment mining for bangla text using support vector machine," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–5.

[8] N. Irtiza Tripto and M. Eunus Ali, "Detecting multilabel sentiment and emotions from bangla youtube comments," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–6.

[9] A. Hassan, N. Mohammed, and A. K. A. Azad, "Sentiment analysis on bangla and romanized bangla text (brbt) using deep recurrent models," 10 2016.

[10] S. Arafin Mahtab, N. Islam, and M. Mahfuzur Rahaman, "Sentiment analysis on bangladesh cricket with support vector machine," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–4.

[11] I. J. Sagina, "Why go large with data for deep learning?" Apr 2018. [Online]. Available: https://towardsdatascience.com/why-go-large-with-data-for-deep-learning-12eee16f708

[12] Stopwords-Iso, "stopwords-iso/stopwords-bn," Oct 2016. [Online]. Available: https://github.com/stopwords-iso/stopwords-bn

[13] D. L. Lee, H. Chuang, and K. Seamons, "Document ranking and the vector-space model," *IEEE software*, vol. 14, no. 2, pp. 67–75, 1997.

[14] H. Shirani-Mehr, "Applications of deep learning to sentiment analysis of movie reviews," in *Technical report*. Stanford University, 2014.

[15] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

[16] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[17] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *international conference on machine learning*, 2016, pp. 2217–2225.

[18] R. A. Dunne and N. A. Campbell, "On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function," in *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*, vol. 181. Citeseer, 1997, p. 185.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Annals of operations research*, vol. 134, no. 1, pp. 19–67, 2005.

[21] M. S. Hossain, S. Rahaman, A.-L. Kor, K. Andersson, and C. Pattinson, "A belief rule based expert system for datacenter pue prediction under uncertainty," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 2, pp. 140–153, 2017.

[22] R. Karim, K. Andersson, M. S. Hossain, M. J. Uddin, and M. P. Meah, "A belief rule based expert system to assess clinical bronchopneumonia suspicion," in *2016 Future Technologies Conference (FTC)*. IEEE, 2016, pp. 655–660.

[23] M. S. Hossain, M. S. Khalid, S. Akter, and S. Dey, "A belief rule-based expert system to diagnose influenza," in *2014 9Th international forum on strategic technology (IFOST)*. IEEE, 2014, pp. 113–116.

[24] R. Ul Islam, K. Andersson, and M. S. Hossain, "A web based belief rule based expert system to predict flood," in *Proceedings of the 17th International conference on information integration and web-based applications & services*. ACM, 2015, p. 3.

[25] M. S. Hossain, S. Rahaman, R. Mustafa, and K. Andersson, "A belief rule-based expert system to assess suspicion of acute coronary syndrome (acs) under uncertainty," *Soft Computing*, vol. 22, no. 22, pp. 7571–7586, 2018.

Bangla Numeral Recognition from Speech Signal Using Convolutional Neural Network

M. I. R. Shuvo
Dept. of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh
insan_shuvo@cse.kuet.ac.bd

Shaikh Akib Shahriyar
Dept. of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh
akib.shahriyar@cse.kuet.ac.bd

M. A. H. Akhand
Dept. of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh
akhand@cse.kuet.ac.bd

***Abstract*— Speech recognition is a process where an acoustic signal is converted to text or words or commands and recognizing the speech. In this paper, a Bangla numeral recognition system from the speech signal is developed utilizing Convolutional Neural Network (CNN). In the proposed system, a speech dataset of ten isolated Bangla digits has been developed consists of 6000 utterances (5 utterances for every 120 speakers) and a feature extraction procedure is performed to elicit significant features from the speech signals using Mel Frequency Cepstrum Coefficient (MFCC) analysis. Then, CNN is trained with the features of the speech signal as input. The efficiency of the proposed system is tested on the dataset developed for this purpose, and acquire 93.65% recognition accuracy. The proposed system is also compared with other existing methods of Bangla numeral speech recognition and outperforms most of the existing systems and proves the superiority of itself.**

Keywords— Speech Recognition,Bangla neumeral, MFCC, Convolutional Neural Netrork.

I. INTRODUCTION

For human communication in real life, speech is the easiest and efficient way. People use other methods of communication rather than speech only when it is not available [1]. As speech is the main medium of communication for human, it can be used as a convenient interface to communicate with computer and other machines. Speech recognition by machines is one of the main research topics in communication between human and machine. Nowadays, research on speech recognition has developed from laboratory demonstrations to real-life applications [2] [3].

Speech recognition is a process where an acoustic signal is converted to text or words or commands and recognizing the speech. In a speech recognition system, computer or machine is able to take speech signals as input and recognize or understand the inputs and produce output in words or text form [4]. Designing of speech recognition systems has reached a new level but there are few problems such as robustness and noise-tolerant recognition which make the systems difficult to use. The speech recognition system can be divided into different types depending on utterance type of speech elements: isolated words, connected words, continuous speech, and spontaneous speech systems. It can also be divided on the basis of speaker mode, vocabulary size, etc. [2] [5]. Here, the pronunciation of the speaker is a vital issue that makes this kind of system speaker-dependent. Different languages differ significantly because each of them has their own phonemes. So, no one can confirm that a recognition system which works well for certain language such as English, will provide a good result for recognition of another language such as Bangla [6].

Research work on speech recognition has been introduced since 1930. But, in case of Bangla speech recognition, it is notable that research on this topic has been started since early 2000 [7] [8]. Many studies in speech recognition systems can be found for different major languages that produce good recognition accuracy for the corresponding language because of the availability of the experimental resources especially proper speech corpus. Bangla is the seventh-ranked language in terms of the number of speakers, but no standard Bangla speech corpus has been found which makes research on Bangla speech recognition more difficult [9].

Different methods have been proposed for recognizing Bangla numeral speech using Artificial Neural Network (ANN), Hidden Markov Model (HMM), Support vector classifier with HMM, Linear Predictive Coding (LPC), Fuzzy logic, etc. Ali et al. proposed a method for isolate Bangla speech recognition using back-propagation neural network. [7]. A study had been investigated for recognizing Bangla isolated word speech using spectral analysis and fuzzy logic. [10]. Muhammad et al. designed a system for Bangla digit automatic speech recognition where features of the speech signals are extracted using MFCC analysis and HMM-based classifiers are used for recognition [9]. Paul et al. proposed a technique for Bangla Speech Recognition System using LPC and ANN [11]. Hasnat et al. proposed a method for isolated and continuous Bangla speech recognition using Mel Frequency Cepstral Coefficients (MFCC) and HMM [8]. Khalil & Mahfuzur developed a system for connected Bangla speech recognition using Artificial Neural Network (ANN). [2]. Kotwal et al. proposed Bangla phoneme recognition

978-1-7281-5242-4/19

method for Automatic Speech Recognition (ASR) using Multilayer Neural Network (MLN) and HMM. [12]. Nahid et al. presented a noble approach to develop an automatic Bangla Real Number recognizer using CMU Sphinx4 [13]. Ali et al. investigated an automatic speech recognition of Bangla words using MFCC, Gaussian Mixture Model (GMM), LPC and DTW [14]. Islam et al. implemented a three-layer backpropagation Neural Network which used as a classifier to recognize Bangla speech [15]. From the review of the previous research work in Bangla speech recognition, we can see that most of the studies use HMM and ANN as the recognizer. But no study uses CNN for recognition.

The main objective of this work to develop a CNN based Bangla numeral speech recognition system. It takes the 2D structured MFCC features as input. The proposed system overcomes the drawbacks which researcher face while working with 1D structured input.

The paper is organized as follows. Necessary preliminaries are described in Section II. Section III describes the dataset description and the proposed system architectures. Section IV explains the experimental results of the proposed method as well as performance comparison with other existing related works. Finally, a short brief of the work is given in the conclusion which is in Section V.

II. PRELIMINARIES

A. Convolutional Neural Network (CNN)

CNN [16], multi-layer variants of ANN are developed to work with two-dimensional data which are well applied to classification and recognition of two-dimensional data by recognizing the patterns directly from input data. CNN architecture comprises an input layer, some convolution-subsampling layers, hidden layer and output layer shown in Fig. 1.

In the convolution process, a small-sized filter known as kernel is applied over the input feature map (IFM) to generate a convolved feature map (CFM) [17]. Weights and bias of a kernel are shared in this way among all positions during convolution process. CFM is calculated from an IMF using Eq. (1).

$$CFM_{x,y} = f(\, b + \sum_{i=1}^{K_h} \sum_{j=1}^{K_w} K_{i,j} * IMF_{x+i,y+j}) \quad (1)$$

Here, $f(\,)$ = Activation function, b = Bias, K_h, K_w = Size of the kernel as $K_h \times K_w$ matrix and * = Two dimensional convolution. In CNN, there is a sub-sampling layer followed by each convolutional layer to simplify the information obtained from the convolutional operation and a feature map known as sub-sampled feature map (SFM) is generated. A sub-sampling operation is shown in Eq. (2). To train CNN, a revised version of Back-Propagation is used [18] [19].

$$SFM_{x,y} = d\,(\sum_{i=0}^{R-1} \sum_{j=0}^{C-1} CFM_{xR-1+i,yC-1+j}) \quad (2)$$

Here, d (.) = Sub-sampling operation on a pooling area.

R, C = Pooling area size.

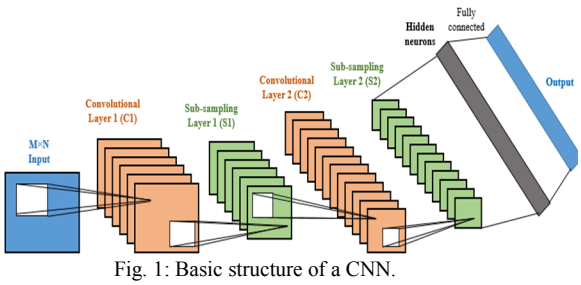


Fig. 1: Basic structure of a CNN.

III. BANGLA NUMERAL RECOGNITION FROM SPEECH SIGNAL USING CNN

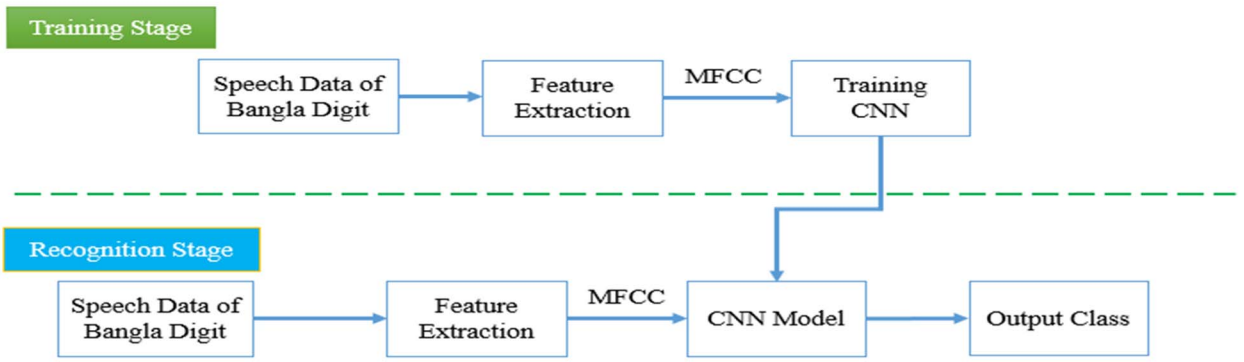
Figure 2(a) demonstrates the basic structure of the proposed speech recognition system. At first, a speech dataset of ten isolated Bangla digits has been developed for this system. In this work, the proposed system has two phases: i) Training stage ii) Recognizing stage. In the training stage, a feature extraction procedure is performed to elicit significant features from the speech signals using Mel Frequency Cepstrum Coefficient (MFCC) analysis. The resultants MFCC features are of numerical form rather than in signal form. Then, CNN is trained with extracted MFCC features. In the recognizing stage, a feature extraction procedure is performed to extract the MFCC features from the numeral speech signals. Then, the MFCC features are fed into the pre-trained CNN model to recognize the numeral speech signals.

A. Dataset Description

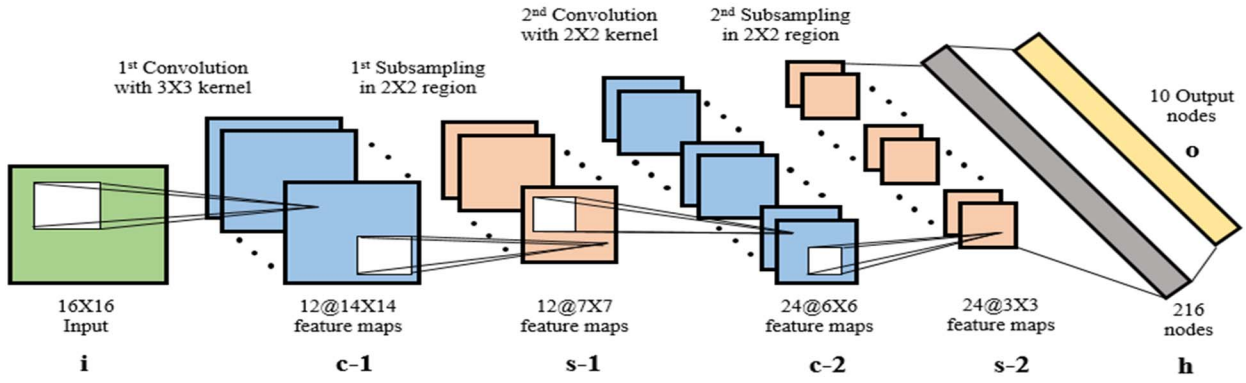
A speech dataset of ten isolated Bangla digits recorded in a noise-free environment from 60 male and 60 female native Bangla speakers from different regions of Bangladesh has been used in this study. This dataset has been developed exclusively for the purpose of this research. The speech signals (utterances) were recorded with a close-talking microphone, voice recording software and high-quality sound card. The recorded utterances had a sampling rate 16 kHz and encoded in 8 bits PCM. Each utterance was one second in length. The number of channel used for voice recording was one and no extra filter was used. Five trails of each digit from each speaker were recoded. By following a ratio of 70:30, training set and testing set were formed. All the five trials for each digit were considered for training and testing sets. So, 2×42×5 instances of each digit were used in the training set and 2×18×5 instances of each digit were used in the testing set.

B. Feature Extraction

The extraction of features from speech signals is an important and common stage for the speech recognition system [18]. Mel Frequency Cepstrum Coefficient (MFCC) analysis is considered in the proposed system to produce meaningful MFCC features for recognition and extract MFCC features from the numeral speech signals using 'librosa' (an open-source library). The number of MFCC filters play a significant role in the MFCC extraction procedure. Here, 16 MFCC filters were used to generate 16×16 features for each numeral speech signal.



(a) Basic structure of the proposed Bangla numeral speech recognition system.



(b) CNN architecture considered in the proposed system.

Fig. 2. Architecture of the proposed Bangla numeral speech recognition system.

C. *Recognition using CNN*

The CNN architecture considered in this study is shown in Fig. 2(b). It consists of two convolution layer c-1 and c-2 with kernel size 3×3 and 2×2, respectively and two subsampling layer s-1 and s-2 with 2×2 local maximum area. 16×16 input features are treated as 256 linear nodes, upon which a simple convolution operation is performed in the input layer i. Finally, all the hidden layer nodes in h are connected to 10 output nodes in o. Here, a particular digit is represented by an output node. During training, total 120 (= ((3×3)+1)×12) parameters for 12 kernels of C1 are updated. In C2, there are total 1176 (= ((12×2×2)+1)×12) parameters for 24 CFMs. Finally, during the training period, the 2160 (= 216×10) weights of the fully connected layer are continuously updated. The training process continues until the desired accuracy is achieved.

IV. Experimental Results and Analysis

The proposed system has been tested rigorously to form the experimental results. The training and test data contain 4200 instances and 1800 instances, respectively. Python, Keras, and Tensorflow are used to implement the proposed system. The experiment was carried out onto a Windows 10 environment on a Desktop with the below configuration: CPU: Intel(R) CoreTM i7-8700K CPU @ 4.70 GHz, RAM: 32 GB, GPU: Nvidia GTX 1050ti 4 GB. To evaluate the performance of the proposed system, the recognition accuracy is considered. The recognition accuracy is calculated using Eq. (3)

TABLE I. CONFUSION MATRIX AND THE DETAILED RECOGNITION ACCURACY OF THE BANGLA NUMERAL SPEECH TEST SAMPLES AFTER 300 ITERATIONS.

Bengali Speech Numeral Digits											
Bengali Numeral	০	১	২	৩	৪	৫	৬	৭	৮	৯	Recognition Accuracy (%)
০	174	0	0	0	0	0	0	0	0	6	96.67
১	0	178	0	0	0	0	0	0	2	0	98.89
২	0	0	176	0	0	0	1	0	0	3	97.78
৩	3	0	0	175	0	0	0	0	0	2	97.23
৪	0	0	0	0	177	2	0	0	1	0	98.34
৫	0	0	0	0	5	172	0	3	0	0	95.78
৬	0	0	5	0	0	0	149	0	0	26	82.78
৭	0	0	0	0	0	2	0	151	27	0	83.88
৮	0	0	0	0	0	2	0	26	152	0	84.45
৯	4	0	2	0	0	0	29	0	0	145	80.56
Avg. Recognition Accuracy											93.64

TABLE II. COMPARATIVE RESULT ANALYSIS OF THE TEST SET RECOGNITION ACCURACY OF THE PROPOSED METHOD WITH OTHER EXISTING METHODS OF BANGLA NUMERAL SPEECH RECOGNITION.

Work reference and Year	Feature Extraction	Recognition	Dataset; Training and Test Instances	Recog. Acc. (Test Set)
Khalil and Mahfuzur [2], 2016	MFCC	ANN	Self-Prepared; 260	89.87%
Ali et al. [7] , 2013	MFCC	ANN	Self-Prepared; 150 and 150	92%
Muhammad et al. [9], 2009	MFCC	HMM	Self-Prepared; 3700 and 1300	93.50 %
Paul et al. [11], 2009	Durbin's recursive algorithm	LPC and ANN	Self-Prepared; 600 and 100	Satisfactory
Proposed Scheme	MFCC	CNN	Self-Prepared; 4200 and 1800	**93.65%**

$$Recognition\ Accuracy = \frac{Correctly\ recognized\ digits}{Recognized+Unrecognized\ digits} * 100\% \quad (3)$$

Table I shows the confusion matrix and the recognition accuracy for all the 10 digits of the test sample after 300 iterations. From the table, it can be seen that the recognition accuracy for the first six digits (0-5) is quite satisfactory. But due to pronunciation complexity, the accuracy of the last four is not high as the first six digits. Two most confusing digit pairs can be found from Table I. One pair includes '৬' and '৯' and another pair includes '৭' and '৮'.

It can be seen from Table I that 14.45% '৬' is misclassified as '৯' and 16.11% '৯' is misclasssifed as '৬'. Similarly, 15% '৭' is misclassified as '৮' and 14.45% '৮' is misclassified as '৭'. This happens because they are phonetically very close to their counterparts when they are pronounced. The proposed system finds satisfactory performance for the digits which do not conflict with other digits at the time of their pronunciation e.g. recognition accuracy for '১' is 98.89%, '২' is 97.78 %, '৩' is 97.23% which are quite different at the time of their pronunciation.

Table II shows the comparative result analysis of the test set recognition accuracy of the proposed scheme with other existing methods of Bangla numeral speech recognition. Table II depicts that, the proposed system achieved test set recognition accuracy of 93.65% where the test set recognition accuracy for the works of [2], [7] and [9] are 89.87%, 92%, and 93.50%, respectively. So, with 93.65% recognition accuracy, proposed method outperforms most of the existing systems and proves the superiority of itself.

V. CONCLUSIONS

This paper presents a novel Convolutional Neural Network (CNN) based system for Bangla numeral recognition from the speech signal. The proposed system has been tested on a self-prepared Bangla numeral Dataset because of the unavailability of a standard Bangla numeral Dataset. The recognition accuracy achieved by the proposed system is compared to some other existing methods and it is observed that this system shows a better performance compared to other prominent methods.

REFERENCES

[1] B. Rehman, Z.Halim, G. Abbas and T. Muhammad, "Artificial Neural Network-Based Speech Recognition Using Dwt Analysis Applied On Isolated Words From Oriental Languages," *Malaysian Journal of Computer Science*, vol. 28(3), pp. 242-262, September 2015.

[2] K. Ahammad and M. Rahman, "Connected Bangla Speech Recognition using Artificial Neural Network," *International Journal of Computer Applications*, vol. 149(9), pp. 38-41, September 2016.

[3] W. Gevaert, G. Tsenov and V. Mladenov, "Neural networks used for speech recognition," J*ournal of Automatic Control*, vol. 20(1), pp. 1-7, January 2010.

[4] Rubi and C. Rana, "A review: speech recognition with deep learning methods," *International Journal of Computer Science and Mobile Computing*, vol. 4(5), pp. 1017-1024, May 2015.

[5] P. Kurzekar, R. R. Deshmukh, V. Waghmare and P. P. Shrishrimal, "Continuous Speech Recognition System: A Review," *Asian Journal of Computer Science and Information Technology*, vol. 4(6), pp. 62-66, June 2014.

[6] S. A. Hossain, M. L. Rahman, F. Ahmed and M. Dewan, "Bangla speech synthesis, analysis, and recognition: an overview", *Proc. NCCPB*, Dhaka, Bangladesh 2004.

[7] Md. A. Hossain, Md. M. Rahman, U. K. Prodhan and Md. F. Khan "Implementation of Back-Propagation Neural Network for Isolated Bangla Speech Recognition", *International Journal of Information Sciences and Techniques (IJIST)*, vol.3(4), July 2013.

[8] M. Hasnat, J. Mowla, and M. Khan, "Isolated and continuous bangla speech recognition: implementation, performance and application perspective", 2007.

[9] G. Muhammad, Y. Alotaibi, and Md. N. Huda, "Automatic speech recognition for Bangla digits," *12th International Conference on Computers and Information Technology (ICCIT)*, pp. 1-5, December 2009.

[10] A. Firoze, M. S.Arifin, R. Quadir and R. Rahman, "Bangla Isolated Word Speech Recognition," *Proceedings of the 13th International Conference on Enterprise Information Systems*, vol. 2 , pp. 73-82, January 2011.

[11] A. K. Paul, D. Das, and M. M. Kamal, "Bangla Speech Recognition System Using LPC and ANN," *Seventh International Conference on Advances in Pattern Recognition*, pp. 171-174, February 2009.

[12] M. R. A. Kotwal, Md. S. Hossain, F. Hassan, G. Muhammad, Md. N. Huda and C. M. Rahman, "Bangla phoneme recognition using hybrid features," *International Conference on Electrical & Computer Engineering (ICECE)*, pp. 1-4, December 2010.

[13] Md. A. Islam, Md. S. Islam and Md. M. H. Nahid, "A Noble Approach for Recognizing Bangla Real Number Automatically Using CMU Sphinx4," *5th International Conference on Informatics, Electronics and Vision (ICIEV)*, pp. 1-6, May 2016.

[14] Md. A. Ali, M. Hossain and M. N. Bhuiyan, "Automatic Speech Recognition Technique for Bangla Words," *International Journal of Advanced Science and Technology*, vol. 50, pp. 51-60, January 2013.

[15] M. R. Islam, A. S. M. Sohail, M. W. H. M. Sadid and A. Mottalib, " Bangla Speech Recognition using three layer Back-Propagation Neural Network", *Proc. NCCPB*, Dhaka, Bangladesh 2005.

[16] T. Liu, S. Fang, Y. Zhao, P. Wang and J. Zhang., "Implementation of Training Convolutional Neural Networks", *arXiv preprint arXiv:1506.01195*, 2015.

[17] "Feature extraction using convolution, UFLDL Tutorial". *Available: http://deeplearning.stanford.edu/,* accessed November 12, 2015.

[18] L. Rabiner, B. H. Juang and B. Yegnanarayana,"Fundamentals of Speech Recognition", *Pearson Education India*, 2008.

End to End Parts of Speech Tagging and Named Entity Recognition in Bangla Language

Jillur Rahman Saurav*, Summit Haque†, Farida Chowdhury*
* Search Engine Pipilika
*†Department of Computer Science & Engineering
*†Shahjalal University of Science & Technology
Sylhet, Bangladesh
{sauravsust71, summit.haque, deeba.bd}@gmail.com

***Abstract*—Automatic Parts of Speech(POS) tagging is one of the most fundamental tasks for a language in Natural Language Processing(NLP), which acts as a feature for solving advanced NLP tasks. Named Entity Recognition(NER) is another essential task of NLP for information retrieval. Researchers could not find up to the mark solution yet on these two tasks for Bangla language compared to other languages, for instance, English, German. Moreover, many solutions heavily depend on handcrafted features that require strong linguistic expertise. As these two sequence labeling tasks are similar, In this work, two different datasets of POS tagging and NER were prepared, and different deep neural network approaches studied for solving these two tasks separately. All of the approaches were end to end and did not need any handcrafted feature like word suffixes or affixes, gazetteers, dictionary. This study came up with an end to end solution using deep neural network-based model consisting of Bi-directional Long short-term memory(BLSTM), Convolutional Neural Network(CNN) and Conditional Random Field(CRF). The proposed model trained on respected datasets achieved an accuracy of 93.86% on POS tagging and a strict f1 score of 0.6285 on NER on prepared datasets, respectively.**

***Index Terms*—Parts of Speech tagging, Named entity recognition, Bangla POS tagging, Bangla NER, Deep neural network, CNN, LSTM, BLSTM, CRF**

I. Introduction

In this era of technology, using NLP and Computer vision, machines are taught to mimic humans on different tasks. Even machines are now playing the role of the personal assistant. Different approaches are studied to make a machine capable of understanding human behavior, interactions.

One goal of NLP is to help the machine to understand human language and respond accordingly naturally. There are some fundamental things of a language, machines needed to learn for developing more advance Artifical Intelligent based system in that language. Extracted POS from given text is one of them. Recognizing a named entity is another necessary task of NLP to perform information retrieval. Solving these kinds of fundamental tasks will help to develop a more advanced system like a chatbot, optimize search results, etc.

In resource-rich languages on the perspective of NLP, researchers have made significant improvements in solving those tasks. For POS tagging in the English language, they have achieved over 97% accuracy [1].

Such kind of benchmark works on primary tasks of NLP has not been found for Bangla, the 7th most spoken language of the world [2].

In this work, we have experimented different approaches for automatic POS tagging and NER in Bangla language. These two tasks are similar on the perspective of machine because it needs to predict a tag for each token of a given text whether a POS tag or NER tag. We have studied end to end approaches where handcrafted features are not necessary during training and serving. We have explored different Deep Neural network models(DNN) as these can learn the non-linearities and now widely been used for almost all pattern recognition and machine learning application and achieving state-of-the-art performances in many sectors [3]. We have used word embeddings as in recent, NLP researchers found that using word embeddings or vector representation of words brings significant increases in the performance [4]. In recent studies, it is found that many problems can be solved by end to end approaches with more data and complex deep neural networks, for instance, end to end speech recognition [5] that motivated us to try end to end approach for these tasks.

Like other resourceful languages, for Bangla language, we can not find so many publicly available datasets to study. So we have prepared our datasets maintaining standards for conducting our research. The making of datasets is briefly described in Section III dataset preparations.

We came up with a model based on BLSTM, CNN and CRF, trained and tested respectively on the two datasets we have prepared for two tasks, outperformed other models on both POS tagging and NER.

The rest of the paper is organized as follows. Section II describes the related work. Section III presents the Datasets used

978-1-7281-5242-4/19

in this work. Section IV explains the deep neural networks that we have used in this work. Section V describes the training process and hyperparameters. The results are discussed in section VI, and we conclude in sectionVII

II. Related Work

More works have been done on POS tagging compared to NER in Bangla language. Some notable previous works on NER and POS tagging are presented below.

A. Named Entity Recognition

In Bangla language, most of the works were done by Ekbal et al. [6]- [13]. They showed the use of CRFs, Maximum Entropy(ME), Support Vector Machine(SVM)s and achieved f1 measure varies from 82% to 91% on the different number of entity types. A resource-based study was done by Chaudhuri et al. [14] using a dictionary, rules, and n-gram based statistical modeling. The reported accuracy by K. S. Hasan et al. is 71.9% on three entity types. The baseline model used by them was CRF.

B. POS recognition

Many experiments have been conducted in the area of POS recognition in Bangla language. These studies attempted to define tag sets and application of different statistical and machine learning model for automatic POS tagging.

CRF based system is featured with word suffixes with lexicons, and NER was proposed by this work [16] for pos tagging. They use 26 tags and a corpus containing 72,341 tokens and achieved 90.3 % accuracy. Another study was done on the same corpus with the Hidden Markov model(HMM) and ME based models [17]. They showed that the ME model beats the HMM model by a margin of 7.5 % more accuracy achieving 88.1% accuracy. In [18], they applied SVM and showed that SVM outperformed the previous CRFs, HMMS, and ME based system for the same corpus. An unsupervised approach for recognizing POS in resource-scarce language was proposed by [19]. For a dataset containing ten tags, they achieved an F1 score of 79 %.

In [20], the authors studied different algorithm on a corpus containing 4000 sentences(tagset not sure). They showed that Global Linear Model (GLM) outperformed HMMs, SVM, CRFs, ME obtaining an accuracy of 93.12%.

A deep neural network approach was proposed by this work [21]. They used Bi-directional LSTM based model with CRF layer on top. They got an accuracy of 86% on the dataset prepared by this work [22].

III. Dataset Preparation

A. NER Dataset

We have prepared our own NER dataset for this research. We have collected news articles from different online Bangla news portals. We picked sentences and tokenized them for tagging purpose. There are many standard schemes like IOB, IOBES, BIO for tagging Named entity. We have selected BIO scheme for our tagging system as a recent study found that it performs better than other tagging schemes [24]. In this scheme, every entity token starts with a B-Tag, and if the entity consisting of more than one tokens, the followings are tagged with I-Tag. This dataset contains four types of entities as the CoNLL-2003 Shared Task [25]. These are person(PER), location(LOC), organization(ORG) and miscellaneous(MISC). Tokens do not belong to these four entity types are considered as other(O). We have followed the tagging guidelines as a study told about what to annotate [26]. We have used a crowdsourcing platform[1] to tag our dataset. We found that this dataset is not the gold standard. The dataset contains 10290 sentences and 176029 tokens. The statistics and tag distribution of the dataset are given in table I and figure I, respectively. We have excluded the other tag in the graphical representation.

Total Sentences	10290
Total Tokens	176029
Total Tags	4

TABLE I
NER DATASET STATISTICS

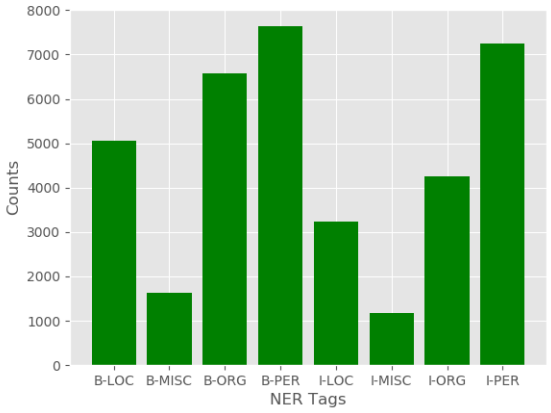


Fig. 1. NER Tag Distributions

B. POS Dataset

We have prepared our dataset for POS tagging. We have used top-level categories of the tagset proposed by the Bureau of Indian Standard (BIS) [23]. We have collected various articles from different types comprising politics, economics, entertainment, sports, lifestyle, etc. from different Bangla online newspapers. We have tokenized the sentences by following standard tokenization scheme. We have tagged 47594 tokens of 4944 sentences. We have included the corpus statistics and the POS tag distribution in table II and figure II We have found the most frequent tag in our corpus is the tag 'Verb' and it occurred in 12251 tokens.

[1] crowd.pipilika.com

Total Sentence	4944
Total Tokens	47615
Total Tags	12

TABLE II
POS DATASET STATISTICS

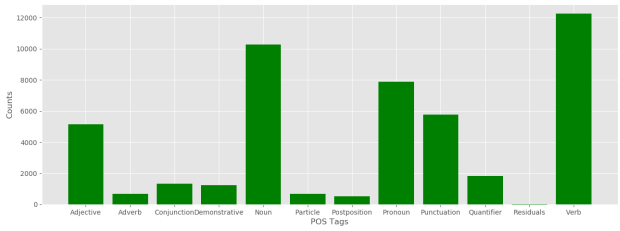


Fig. 2. POS Tag Distributions

IV. Neural Network Architecture

We have used traditional deep neural networks CNN, LSTM, BLSTM in our experiments. Brief descriptions of the models that we have used are given below.

A. CNN

In recent studies found that the CNN is not only useful for computer vision but also very much effective in extracting morphological information a word [27], which can be used for propagating character representations of words into neural networks.

B. LSTM

RNNs are very efficient in capturing long-distance dependencies, and one of its variant LSTM solved its vanishing gradient problem [28] - [30]. LSTM cells carry past information through themselves. Each cell has different gates to interact with the data passing through it. LSTMs cell can update, remove the portion of data using the gates. A schematic design of a basic LSTM cell is given in figure III.

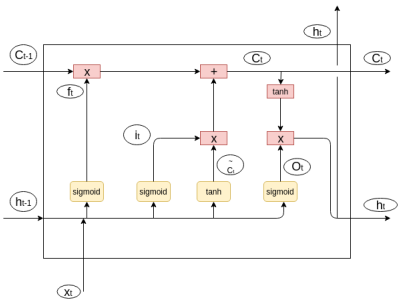


Fig. 3. Basic LSTM cell

The equations used by an LSTM cell to perform operations are given below.

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \quad (1)$$
$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \quad (2)$$
$$C_t^{\sim} = tanh(W_c.[h_{t-1}, x_t] + b_c) \quad (3)$$
$$C_t = f_t * C_{t-1} + i_t * C_t^{\sim} \quad (4)$$
$$o_t = \sigma(W_o.[h_{t1}, x_t] + b_o) \quad (5)$$
$$h_t = o_t * tanh(C_t) \quad (6)$$

Here σ symbolizes element-wise sigmoid operation. And . shows the element-wise product. C_t and x_t stand for the context vector and the input vector at time t

C. BLSTM

Generally, LSTM cells only have information about the past context. Its hidden states do not have any information about the future context. Dyer et al. proposed an elegant solution which uses the sequence forwards and backward to two different states and concatenates them to form the final output [31] which is Bi-directional LSTM(BLSTM)

D. CRF

Contemplation of the correlation between neighborhood labels and jointly decode the best chain of labels for a stated input sentence is useful for general structured prediction or sequence labeling tasks [32]. A coherent instance is- in the case of POS tagging, an adjective presumably followed by a noun than a verb, as well as I-ORG, refrain following I-PER on NER with standard BIO annotation.

V. Models, Training & Hyperparameters

We have tried five models on both tasks. The first three of them were CRF on top of BLSTM. The first one was without pre-trained embedding and character embedding. The second model was without character embedding. The third one included both character embedding and pre-trained embedding. The fourth model we have tried CNN on top of BLSTM with character embedding, and pre-trained embedding included. The final one we have used was CNN on top character embedding and BLSTM on top of the concatenation of word and character embedding after applying CNN on it. The architecture of our BLSTM-CNN-CRF model is given in figure IV.

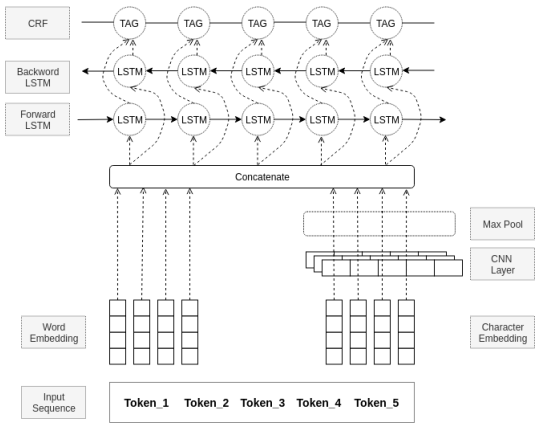


Fig. 4. BLSTM-CNN-CRF Model Architecture

We have used keras[2] for all of our training and testing. Different hyperparameter tunings were evaluated. We have tried LSTM with units 100, 75, 50 with different recurrent dropout. We have tried different kernel size for CNN and found that kernel size 5 did better. We have also tried different optimization techniques like Nadam [33], Adam [34], RMSProp [35] and found that Adam performed faster and better. As the dropout technique heavily used to reduce overfitting in deep neural networks [36], we have used it with different rate and found that dropout rate 0.5 performed best. This work [37] developed the pre-trained word embedding model we have used. We have used scikit-learn[3] train-test split API to split the dataset into train and test set and the ratio was 80 to 20. Important hyperparameters are given in the table III.

params	best performance
LSTM units	100
Char embedding size	30
CNN kernel, filters	5, 30
Optimizer	adam
Dropout rate	0.5

TABLE III
PARAMETERS AND HYPERPARAMETERS

VI. RESULTS & ANALYSIS

A. NER

We have used the strict f1 score as a metric for evaluation where predicted tokens of an entity must need to exact same of the true label. Partial matching of an entity is considered as a wrong prediction. We have used seqeval, an implementation of Standard NER evaluation[4]. We have found the best result for BLSTM-CNN-CRF model with a strict f1 score of .6284. We have also found that our model performs better on recognizing the most frequent entity the person entity and performs worst on identifying the least occurred tag in the dataset. The performances of the models and the result achieved by the models we have used described in table IV and table V respectively.

model	f1
BLSTM-CRF[without ce, without pwe]	47.23
BLSTM-CRF [without ce]	59.22
BLSTM-CRF	62.20
BLSTM-CNN	60.40
BLSTM-CNN-CRF	**62.84**

TABLE IV
STRICT F1 SCORES OBTAINED BY THE MODELS ON NER DATASET

B. POS

Experimenting on the POS dataset, we have found that, like before our BLSTM-CNN-CRF based model best with an accuracy of 93.50%. As expected, it correctly detected all the punctuations of the test case. The model performances and the precision, recall, and f1 measures found from BLSTM-CNN-CRF model are given in tableVI and table VII respectively.

[2]keras.io
[3]http://scikit-learn.org/
[4]https://github.com/chakki-works/seqeval

tag	precision	recall	f1	support
LOC	0.5860	0.5402	0.5621	896
PER	0.8067	0.7759	0.7910	1178
ORG	0.5890	0.5788	0.5838	1161
MISC	0.3898	0.1631	0.2300	282
micro avg	0.6576	0.6016	0.6284	3517
macro avg	0.6452	0.6016	0.6193	3517

TABLE V
BLSTM-CNN-CRF'S CLASSIFICATION REPORT ON NER DATASET

model	accuracy
BLSTM-CRF [without ce, without pwe]	90.04
BLSTM-CRF [without ce]	91.70
BLSTM-CRF	92.29
BLSTM-CNN	92.50
BLSTM-CNN-CRF	**93.86**

TABLE VI
ACCURACIES OBTAINED BY THE MODELS ON POS DATASET

tag	precision	recall	f1	support
Noun	0.6491	0.6677	0.6583	1252
Postposition	0.7000	0.5957	0.6437	94
Adjective	0.5896	0.5972	0.5934	854
Adverb	0.7573	0.5735	0.6527	136
Punctuation	1.0000	0.9991	0.9996	1150
Verb	0.7426	0.7233	0.7328	1388
Pronoun	0.6729	0.6232	0.6471	1205
Quantifier	0.8799	0.9071	0.8933	323
Conjunction	0.9048	0.8736	0.8889	261
Demonstrative	0.8091	0.7807	0.7946	228
Particle	0.9280	0.8056	0.8625	144
Residuals	1.0000	0.7500	0.8571	8
micro avg	0.7556	0.7390	0.7472	7043
macro avg	0.7558	0.7390	0.7468	7043

TABLE VII
BLSTM-CNN-CRF'S CLASSIFICATION REPORT ON POS DATASET

VII. CONCLUSION

In this work, we have experimented different deep neural networks with varying parameters for solving two similar tasks POS tagging and NER in Bangla language. We have come up with BLSTM-CNN-CRF based truly end to end solution for solving these two sequence labeling tasks. For this, we have collected news articles from different categories of different Bangla online news portals and prepared two datasets using standard tags and tagging schemes. We have used BIS top POS categories for making POS tagging dataset and used BIO tagging scheme for tagging named entity. For lackings of publicly available dataset regarding these task for Bangla

language, we can not compare our work with others. But we have evaluated our models on our datasets using standard train test split. Our proposed model did better in both categories. In POS tagging, our proposed model achieved an accuracy of 93.86%, and in NER, our model achieved an f1 score of .6284. This work is several directions for future research. One can try to solve the tasks without being heavily dependent on handcrafted features that require domain-specific expertise. Sentence embeddings and more complex neural networks can be applied to large datasets to achieve better performance.

ACKNOWLEDGMENT

This research work is partially funded by Access to Information (a2i) programme ran from the Prime Minister's office of Bangladesh.

REFERENCES

[1] Manning, C.D., 2011, February. Part-of-speech tagging from 97% to 100%: is it time for some linguistics?. In International conference on intelligent text processing and computational linguistics (pp. 171-189). Springer, Berlin, Heidelberg.

[2] M. Paul Lewis, Gary F. Simons, Charles D. Fennig, "Ethnologue: Languages of the World" in Nineteenth, Dallas, Texas:SIL International, 2016.

[3] LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. nature, 521(7553), p.436.

[4] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J., 2013. Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119)

[5] Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G. and Chen, J., 2016, June. Deep speech 2: End-to-end speech recognition in english and mandarin. In International conference on machine learning (pp. 173-182).

[6] A. Ekbal and S. Bandyopadhyay, Named entity recognition using support vector machine: A language independent approach, International Journal of Electrical, Computer, and Systems Engineering, vol. 4, no. 2, pp. 155170, 2010.

[7] A. Ekbal and S. Bandyopadhyay, Bengali named entity recognition using classifier combination, in ICAPR. IEEE, 2009, pp. 259262.

[8] A. Ekbal and S. Bandyopadhyay, Named entity recognition in bengali: A multi-engine approach, Northern European Journal of Language Technology, vol. 1, no. 2, pp. 2658, 2009.

[9] A. Ekbal and S. Bandyopadhyay, A web-based bengali news corpus for named entity recognition, Language Resources and Evaluation, vol. 42, no. 2, pp. 173182, 2008.

[10] A. Ekbal and S. Bandyopadhyay, Development of bengali named entity tagged corpus and its use in ner systems, in Proc. of the 6th Workshop on Asian Language Resources, 2008.

[11] A. Ekbal and S. Bandyopadhyay, Bengali named entity recognition using support vector machine, in Proc. of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages, 2008.

[12] A. Ekbal, R. Haque, and S. Bandyopadhyay, Named entity recognition in bengali: A conditional random field approach, in Proc. of the 3rd Joint Conference on NLP, 2008.

[13] A. Ekbal and S. Bandyopadhyay, A hidden markov model based named entity recognition system: Bengali and hindi as case studies, in International Conference on PRML. Springer, 2007, pp. 545552.

[14] B. B. Chaudhuri and S. Bhattacharya, An experiment on automatic detection of named entities in bangla, in Proc. of the IJCNLP-08 Workshop on Named Entity Recognition for South and South East Asian Languages, 2008.

[15] K. S. Hasan, V. Ng et al., Learning-based named entity recognition for morphologically-rich, resource-scarce languages, in Proc. of the 12th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2009, pp. 354362.

[16] Ekbal, A., Haque, R. and Bandyopadhyay, S., 2007, December. Bengali part of speech tagging using conditional random field. In Proceedings of Seventh International Symposium on Natural Language Processing (SNLP2007) (pp. 131-136).

[17] Ekbal, A., Haque, R. and Bandyopadhyay, S., 2008. Maximum entropy based bengali part of speech tagging. A. Gelbukh (Ed.), Advances in Natural Language Processing and Applications, Research in Computing Science (RCS) Journal, 33, pp.67-78.

[18] Ekbal, A. and Bandyopadhyay, S., 2008, December. Part of speech tagging in bengali using support vector machine. In 2008 International Conference on Information Technology (pp. 106-111). IEEE.

[19] Dasgupta, S. and Ng, V., 2007, June. Unsupervised part-of-speech acquisition for resource-scarce languages. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL) (pp. 218-227).

[20] Mukherjee, S. and Mandal, S.K.D., 2013, December. Bengali parts-of-speech tagging using global linear model. In 2013 Annual IEEE India Conference (INDICON) (pp. 1-4). IEEE.

[21] Alam, F., Chowdhury, S.A. and Noori, S.R.H., 2016, December. Bidirectional lstmscrfs networks for bangla pos tagging. In 2016 19th International Conference on Computer and Information Technology (ICCIT) (pp. 377-382). IEEE.

[22] Bali, M.K. and Biswas, P., 2010. Indian language part-of-speech tagset: Bengali ldc2010t16. In Philadelphia: Linguistic Data Consortium, Tech. Rep..

[23] Dash, N.S., 2013. Part-of-Speech (POS) Tagging in Bengali Written Text Corpus. International Journal on Linguistics and Language Technology, 1(1), pp.53-96.

[24] Reimers, N. and Gurevych, I., 2017. Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. arXiv preprint arXiv:1707.06799.

[25] Sang, E.F. and De Meulder, F., 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. arXiv preprint cs/0306050.

[26] Fort, K., Ehrmann, M. and Nazarenko, A., 2009, August. Towards a methodology for named entities annotation. In Proceedings of the Third Linguistic Annotation Workshop (pp. 142-145). Association for Computational Linguistics.

[27] Yin, W., Kann, K., Yu, M. and Schtze, H., 2017. Comparative study of CNN and RNN for natural language processing. arXiv preprint arXiv:1702.01923.

[28] Bengio, Y., Simard, P. and Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), pp.157-166.

[29] Gers, Felix A., Jrgen Schmidhuber, and Fred Cummins. "Learning to forget: Continual prediction with LSTM." (1999): 850-855.

[30] Hochreiter, Sepp, and Jrgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[31] Dyer, C., Ballesteros, M., Ling, W., Matthews, A. and Smith, N.A., 2015. Transition-based dependency parsing with stack long short-term memory. arXiv preprint arXiv:1505.08075.

[32] Lafferty, J., McCallum, A. and Pereira, F.C., 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

[33] Dozat, T., 2016. Incorporating nesterov momentum into adam.

[34] Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[35] Tieleman, T. and Hinton, G., 2012. Lecture 6.5-rmsprop, coursera: Neural networks for machine learning. University of Toronto, Technical Report.

[36] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), pp.1929-1958.

[37] Ahmad, A. and Amin, M.R., 2016, December. Bengali word embeddings and it's application in solving document classification problem. In 2016 19th International Conference on Computer and Information Technology (ICCIT) (pp. 425-430). IEEE.

Towards Lexicon-free Bangla Automatic Speech Recognition System

Md. Mehadi Hasan*, Md. Ariful Islam†, Shafkat Kibria‡ and Mohammad Shahidur Rahman§
Department of Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet-3114, Bangladesh
Email: *mehadi541@gmail.com, †arif.islam.j@gmail.com, ‡shafkat80@gmail.com, §rahmanms.bd@gmail.com

***Abstract*—This article presents a lexicon-free Automatic Speech Recognition (ASR) system for the Bangla language and investigates an open-source large Bangla ASR corpus, which proved by OpenSLR. The model has been trained using improved MFCC acoustic features with a deep LSTM as an acoustic model. We have tried two types of decoding techniques in the decoding or the last part of the ASR; one is using a joint decoder of Connectionist Temporal Classification (CTC) and a statistical Language Model (LM) for beam decoding, and another is CTC based greedy decoding. We have trained and investigated the performance of our ASR with non-augmented speech as an input. The achieved results are outstanding compares to the results obtained from past researches that have used the End-to-End approaches for Bangla ASR. On the test dataset, our End-to-End system has obtained different results using two distinct decoders. The obtained results are 39.61% WER and 18.50% CER using the greedy decoder and 27.89% WER and 12.31% CER, which are a little bit improved results, using the beam decoder. This achievement is state of the art for continuous Bangla ASR.**

***Index Terms*—Speech Recognition, CTC, WER, CER, LSTM, lexicon-free Bangla ASR, OpenSLR**

I. Introduction

Speech is the vocalized form of communication used by humans and some animals, which is based upon the syntactic combination of items drawn from the lexicon or the phoneme inventory. Humans can manage the out of vocabulary (OOV) words particularly the noun or name words from the phoneme inventory. The Automatic Speech Recognition (ASR) system is also built on similar concepts – speech recognition based on the lexicon [1]–[14] or the phoneme inventory i.e., lexicon-free that help us to recognize the all vocabulary including the OOV words [15]–[25]. Speech is the easiest and accessible communication medium. It is the most substantial, homely, adaptable methodology. In speech recognition, there can be speech mode, speaker mode, and vocabulary size. In speech recognition, there are isolated speech, connected speech, continuous speech. And in speaker mode, there are speaker depended, speaker independent, speaker adaptive. Also, there is also small, medium and large vocabulary size. As our everyday life is winding up noticeably more counting on technology [2], Speech recognition is a need to interact effortlessly with machines. We know communication between human and computer are limited. Human needs particular skills to use computers. If the human can use their natural language to communicate with machines, then it will be so much easier to anyone to use computers in every sphere of life like searching any information in Google or any other search engine. That is why, ASR system is desirable for all languages.

The Bangla language is a low-resource and challenging language where morphological parsing fairly complex [26], [27]. Though Bangla is the 7th broadly known language [28] in the world, there are several speech recognition researches have been found on Bangla – mostly on lexicon base with limited vocabulary size [1], [2], [4], [7], [10]–[12], [14]. besides, a very few researches have been found on large vocabulary-based continuous speech recognition (LVCSR) [9], [13], [25] for Bangla. The beginning of the ASR research was around in 1930, but for Bangla, the research work has begun around in 2000 [6]. Moreover, the lack of large speech corpora with aligned text and standard lexicon with phonetic transcription for Bangla language is one of the key issues to build a suitable LVSCR system for Bangla. Working with a language like Bangla or Bengali that the availability of parallel data (speech with aligned text and standard pronunciation model or lexicon) is a major problem. Besides, building standard pronunciation model or lexicon is also sophisticated task and need expert involvement. To meet this challenge, we can only emphasize on the speech corpus with aligned text and develop the lexicon-free End-to-End ASR system [8], [16]–[22], where the system can learn the pronunciation model from the aligned text.

Recognizing speech means the technique of translating a speech signal of a voice expression into relating text interpretation, for example, words, phonemes or other dialect items. ASR system [3], [6], [28] converts speech to text so that the system is able to extract semantic meaning from the text. Also, the machine can search automatically using the extracted text from voice. So, now a days ASR has become an essential researches area. However, applying stochastic language model (LM) [3], [5] alongside Gaussian Mixture Models (GMM)-Hidden Markov Model(HMM) [5] hybrid ASR was getting remarkable performance in past

978-1-7281-5242-4/19

decades considering the wake of attempting in Artificial Neural Network (ANN) [5], [6]. However, in 2012, Deep Belief Network - Deep Neural Networks (DBN-DNNs) were introduced as the core technology used to do Acoustic Modeling (AM) replacing the 30-years old standard in the industry of ASR system the GMM-HMM techniques. The DBN-DNNs replace the GMM part of the hybrid ASR, which is DBN-DNNs-HMMs [5], [6], were introduced to do the acoustic modeling by the prominent Speech research groups like – Google, IBM, Microsoft Research, Baidu, etc.From 2015, these prominent research groups have released an even better NN based acoustic model using Long Short-Term Memory (LSTM) based on Recurrent Neural Networks (RNN) with Connectionist Temporal Classification (CTC) loss function and sequence discriminative training techniques [18], [20], [21], [23], [24] – the End-to-End technique that replaces the DNN based hybrid ASR techniques. The End-to-End technique is a lexicon-free ASR and helps learn the pronunciation model from the speech with aligned text. But, there is one of the key issues to approach this technique is – it needs a relatively larger speech corpus than the hybrid ASR techniques. We are expressing our gratitude to the OpenSLR (Open Speech and Language Resources) site, which is provided by Google Inc., has a publicly published large Bengali ASR training data set (containing 196K utterances) [29]. This dataset is adequate enough to investigate the End-to-End technique for lexicon-free Bangla ASR.

This article reports the investigation the OpenSLR's Bangle ASR corpus [29] using the End-to-End technique towards the lexicon-free Bangla ASR and presents some detail about this corpus. There are several types of Seq2Seq models like RNN Transducer [16] Listen, Attend and Spell (LAS) [17] Neural Transducer [18] Monotonic Alignments [19]. Besides Deep LSTM-RNN with CTC loss function is also used in the End-to-End ASR system [8], [20]–[24] for lexicon free acoustic modeling. In this study, the End-to-End method - Deep LSTM-RNN with CTC loss function has been chosen and OpenSLR's Bangle ASR corpus has been used for training and testing our ASR system. The duration of the whole dataset is approximate 250 hours. There are several open source toolkits for hybrid ASR methods i.e. Kaldi, CMUSphinx, HTK etc. and for End-to-End ASR methods i.e. ESPnet, DeepSpeech2 etc. In this research, DeepSpeech2 toolkit, which is an open source Speech-To-Text toolkit from Baidu and used in several Deep Speech research papers [23], [24] has been used for training the lexicon-free ASR system.

II. Related Works

Most of the previous researches of speech recognition for Bangla have been done on small corpus with limited number of vocabulary size. Several research works have been found on the isolated or stop word speech of Bangla with small corpus i.e. [4], [14], [30], [31] and on continuous speech recognition for Bangla with small corpus [1], [4], [10]–[12], [30]; on the other hand, very few research works have been found on LVCSR or large corpus development for Bangla [9], [13], [25], [32]. We can divide the previous research activities on Bnagla speech recognition mainly in two parts - (a) research using old ASR methods like – GMM-HMM, Back Propagation Artificial Neural Network (ANN) etc. (b) research using latest ASR methods like – deep learning LSTM-RNN, RNN-CTC, CNN etc.. These parts of the research activities are explained below -

There several research papers and key achievements from using old ASR methods on Bangla speech corpus with limited vocabulary. Shahena Sultana et al. (a group of researchers from KUET) worked on Continuous Bangla ASR and, in their system, the vocabulary size was 270 Bangla words. The background engine of their ASR system is the Speech Application Programming Interface (SAPI) by Microsoft and achieved 74.81% accuracy for Bangla speech recognition. [10]. Ali Hossain et al. implemented ANN, which is trained with Back Propagation, for recognizing just Bangla digits and performance of their ASR is 96.33% for the known speaker, and the rate degrades to 92% for the unknown speaker [31]. Md. Abul Hasnat et al. of BRAC University did research both on Isolated and Continuous Bangla Speech Recognition. They applied GMM-HMM-based hybrid ASR. Their system has been trained with 100 unique Bangla words and, for stop-word speeches, the recognition accuracy for speaker-dependent and speaker-independent system are 90% and 70%, respectively [4]. Determining uninterrupted speech with ANN classifier has the average performance rate of 73% [1]. Besides, automatic recognition of real numbers was implemented by Md. Mahadi Hasan Nahid et al. using CMU SPHINX and their ASR system had 85% accuracy in personal computer (PC) and 75% accuracy in android mobile [12]. B. Das et al. from IIT Kharagpur have evaluated a Bengali speech corpus with HTK and their corpus named as "SHRUTHI", which is relatively large corpus with 20 hours of speech data. However, M. J. Rahman Saurav et al. also worked on keyword voice search for Pipilika and their corpus contain 500 unique isolated or stop words and 50 speaker's utterances of those words. They had used the KALDI's GMM-HMM recipe and achieved the accuracy 92% on unknown speaker and their corpus contained 5.5 hours of speech data [33].

On the other hand, a few number of research activities have been found that have implemented latest ASR methods. A remarkable research paper for Bangla language [11] with double layered LSTM-RNN, the WER was 13.2% and phoneme detection error rate was 28.7% on Bangla-Real-Number speech corpus and their corpus contains near about 1.5 hours of speech data. Recently, Alif Al Amin et al. have used deep neural network for continuous Bengali ASR system using Kaldi toolkit's recipes [13] and they have used "SHRUTHI" corpus to evaluate the performance in DNN-HMM method. Another recent work that has proposed noise robust End-to-End Bangla ASR system,

where CER for clean speech was 12.31% and for noisy speech was 9.15% CER by augmentating the input speech and 18.42% CER and 53.687% WER on clean speech without any enhancement the input speech [25]. They have used CNN for features extraction and GRU-CTC for End-to-End Bangla ASR system. They have claimed that their GRU-CTC has trained with 350 hours of speech corpus, where 50 hours of speech from Babel dataset [34] and 300 hours from their own internal speech corpus that is not publicly published.

III. System Architecture

In this section, the components of DeepSpeech2, which is used as an End-to-End system in this study, are described. A joint decoder is used here to get better results. DeepSpeech2 is the new promising and well-known lexicon-free models of provided by Baidu research community [23] (see architecture in figure 1)

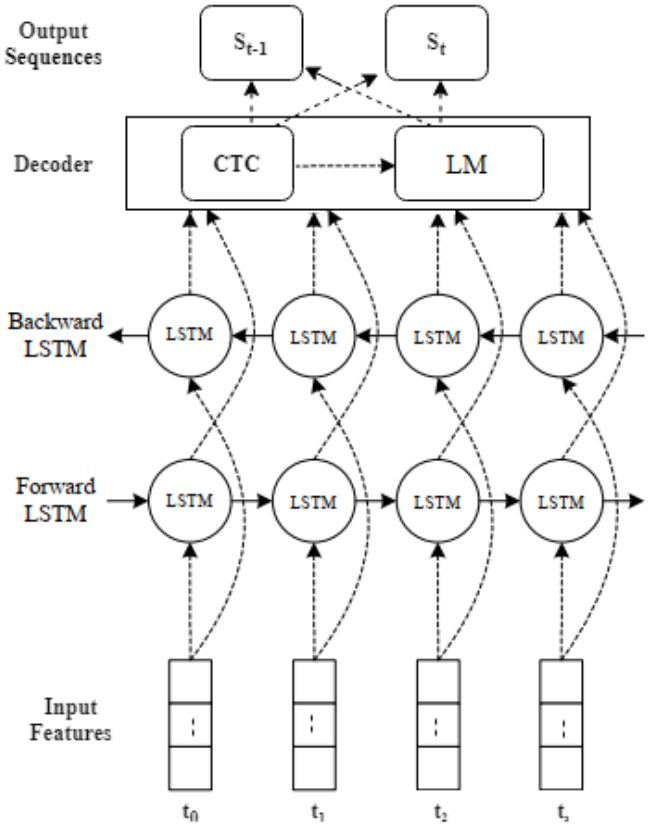


Fig. 1: System architecture

A. Feature Extraction

The remarkable feature extraction process for clean dataset is Mel Frequency Cepstral Coefficient (MFCC) [5]. MFCC can imitate human voices and perceptions that works like our human ear filtering. We have found that 19 MFCC coefficients gave better recognition in our training model; where the feature vectors are 209-dimensional across 11 frames in the 19-dimensional vectors.

At first the 19-dimensional Mel-frequency cepstral coefficient (MFCC) features are spliced in time taking a context size of 11 frames; then linear discriminant analysis (LDA) process is used to do the de-correlation and dimensionality reduction. The output features from the LDA are further de-correlated using maximum likelihood linear transform (MLLT). The MLLT process is followed by feature-space maximum likelihood linear regression (fMLLR) to do the speaker normalization (see figure 2). [35]

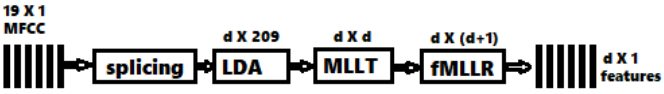


Fig. 2: Improved Output from MFCC on top of LDA + MLLT + fMLLR

B. Acoustic Modeling

In End-to-End technique, speech recognizer encoder namely Acoustic Model (AM) is the deep LSTM network and a new variant of LSTM is Gated Recurrent Unit (GRU). GRU is better than LSTM in the sense of memory efficiency. We have used the bidirectional deep LSTM and GRU network as an acoustic model. It works without lexicon and it passes the output to the decoder. Performance of the deep LSTM network is a little bit better than the GRU network. We have found better results by using 12 hidden layers and 724 hidden units in each layer for the deep LSTM network.

C. Joint Decoding

A warp CTC library for PyTorch backend is used here that is developed by [23], which gives 5-10% speed up in the total training time. Here, Here, a beam search decoder has been used to synchronous with output-label. Throughout decoding, we perform joint decoding by joining both attention-based and CTC scores in beam search algorithm to more uneven alignments. We have used the Bahdanau attention layer in our joint decoder [36]. The distribution of probability for CTC can be calculated using the equation 1 is shown below.

$$P(y|x) = \sum_{\pi \epsilon \phi(y)} P(\pi Vx) \approx \sum_{\pi \epsilon \phi(y)} \prod_{t=1}^{T} P(\pi_t Vx) \quad (1)$$

Score conjunction with attention p^{att} and CTC p^{ctc} log probabilities are performed throughout the beam search where the equation will be following:

$$\log p^{hyb}(y_n|y_{1:n-1}, h_{1:T'}) = \alpha \log p^{ctc}(y_n|y_{1:n-1}, h_{1:T'}) + (1 - \alpha)\log p^{att}(y_n|y_{1:n-1}, h_{1:T'}) \quad (2)$$

where, y_n be a hypothesis of output label at position n given a history $y_{1:n-1}$ and encoder output $\mathbf{h}_{1:T'}$.

The output from the joint decoder has been passed through a 3-gram language model (LM), which has been

built using the SRILM toolkit [37] (here n-gram=3 has been chosen and trained with a text corpus of total 1.6 million words).

IV. Data Preprocessing and Expermental Setup

We have used the OpenSLR's "Large Bengali ASR training dataset" [29] that is available and recently published by Google. The data has some anomalies; so, we have organized the dataset manually and using a python script. The FLAC files have been converted to WAV file where the sampling rate is 16kHz and 16-bit PCM mono. The total duration of the dataset is almost 250 hours. We separated the corpus in train, test and validation folders where each folder has 'txt' and 'wav' named folder. We have kept 70% of data in the training dataset, 10% of data in the test dataset and 20% of data in the validation dataset.

After preprocessing the dataset, we have found some interesting information about the OpenSLR's corpus that are shown in the table I below:

Total Sentences:	2,10,914	Total Duration:	250 hrs
Unique Sentences:	1,07,6,06	Unique Words:	58,564
Speakers:	527	Male=399	Female=128

TABLE I: OpenSLR's Large Bengali ASR Speech Corpus

Several experimental parameters and corpus setup have been configured to train the DeepSpeech2 model, which are shown below –

- Each utterance duration around 2.5 seconds
- Feature Extraction MFCC
- Sampling Rate 16000 Hz
- Vocabulary size 60,000
- (Beam, Batch, Window, Hidden) size 10, 20, 25ms and 724 respectively
- Hidden layers 12
- Window stride 10ms
- Learning rate 1.75e-6
- momentum and learning annealing 0.9, 1.1
- 3-gram language model

V. Experiment and Results Analysis

An open source Bangla speech corpus from OpenSLR which contains the number of total files is 210914 has been used for experiments. The corpus also contains 16kHz sampling rate and 1 channel audio signals which has been given as input for the DeepSpeech2 model. The model has converged after 55 epochs and the optimization algorithm, SGD (Stochastic Gradient Descent) with momentum, minibatch size and the learning rate has been configured (see section IV for configuration setup). The initialized learning rate has annealed by a constant factor 1.1 after each epoch. Two decoders have used here, one is the greedy decoder without LM and another one is beam decoder with LM. The beam decoder has given better results than the greedy one. The maximum frame length has been set as 6500 frames. Total training has taken over 5 days on a double GeForce GTX 1080 Ti GPU in multi-processing.

The LM has been created by the text corpus in order to reduce LM perplexity and the optimum parameters (α, β) values have been set by trial and error basis. We have found the better estimated parameters are: $\alpha = 4.5$, $\beta = 3.2$ and beam width = 100. Furthermore, conversational text may be used for minimizing the Out of Vocabulary (OOV) value. The 3-gram LM is about 1GB in ARPA format and 650MB in binary format. The binary formatted LM has been chosen for faster decoding process.

The performance metrics WER and CER have been calculated for trained model. WER and CER have been decreased as the number of epochs has been increased. The transition is shown in the figure 3 below:

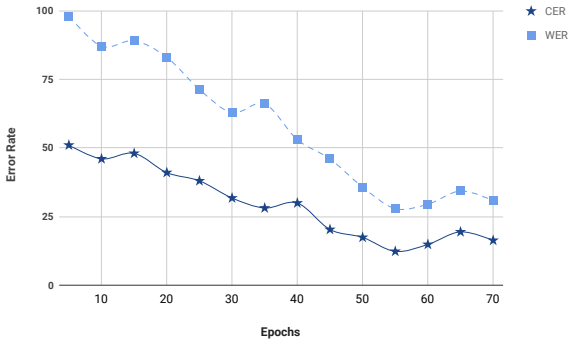


Fig. 3: WER vs CER

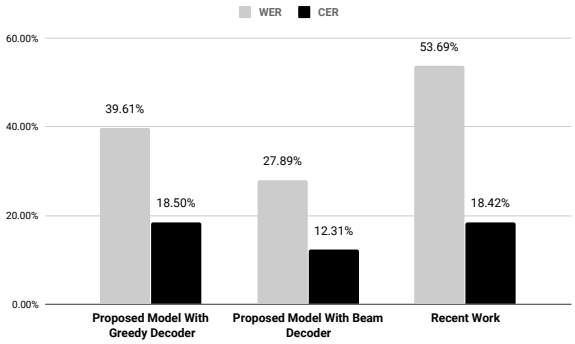


Fig. 4: Comparison with a recent similar work

The results have been compared with the results of a recent research work of GRU-CTC approach on Socian read speech corpus [25]. But they achieved the best accuracy after augmentation the speech and our ASR system's results are good enough without augmentation the speech. So we have only considered their ASR system's results that they achieved without augmentation the speech for fair comparisons. The figure 4 shows the results comparisons and the figure 5 shows some examples output of decoded

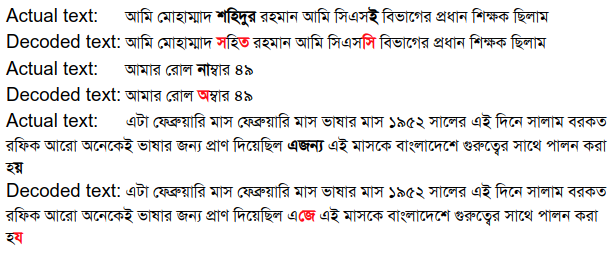


Fig. 5: Sample Decoded Transcription

transcription of some speeches using beam decoder with LM from our model (these examples of speech have been separately recorded by us and are not the part of the OpenSLR's corpus).

VI. Conclusion

In this study, we have examined a lexicon-free End-to-End ASR technique that has used an improved acoustic features - MFCC features on top of LDA + MLLT + fMLLR, deep LSTM and CTC on an open source large Bangla ASR corpus from OpenSLR, which is provided by Google. For evaluation the performance of the test dataset, we have used two types of decoders – one is beam decoder, which is a join decoder using CTC with a 3-gram LM, and another is greedy decoder, which is using CTC. We have achieved 27.89% WER and 12.31% CER using beam decoder and 39.61% WER and 18.50% CER using greedy decoder on the test dataset for the non-augmented speech as an input.

References

[1] K. Rahman, M. Hossain, D. Das, T. Islam, and M. Ali, "Continuous bangla speech recognition system," in *Proc. 6th international conference on computer and information technology (ICCIT03)*, 2003.

[2] R. Karim, M. S. Rahman, and M. Z. Iqbal, "Recognition of spoken letters in bangla," in *Proc. 5th international conference on computer and information technology (ICCIT02)*, 2002.

[3] B.-H. Juang and L. R. Rabiner, "Automatic speech recognition–a brief history of the technology development," *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara*, vol. 1, p. 67, 2005.

[4] M. Hasnat, J. Molwa, and M. Khan, "Isolated and continuous bangla speech recognition: Implementation," *Performance and application perspective*, 2007.

[5] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[6] X. Huang, J. Baker, and R. Reddy, "A historical perspective of speech recognition," *Communications of the ACM*, vol. 57, no. 1, pp. 94–103, 2014.

[7] A. K. Paul, D. Das, and M. M. Kamal, "Bangla speech recognition system using lpc and ann," in *Advances in Pattern Recognition, 2009. ICAPR'09. Seventh International Conference on.* IEEE, 2009, pp. 171–174.

[8] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[9] B. Das, S. Mandal, and P. Mitra, "Bengali speech corpus for continuous auutomatic speech recognition system," in *Speech Database and Assessments (Oriental COCOSDA), 2011 International Conference on.* IEEE, 2011, pp. 51–55.

[10] S. Sultana, M. Akhand, P. K. Das, and M. H. Rahman, "Bangla speech-to-text conversion using sapi," in *Computer and Communication Engineering (ICCCE), 2012 International Conference on.* IEEE, 2012, pp. 385–390.

[11] M. M. H. Nahid, B. Purkaystha, and M. S. Islam, "Bengali speech recognition: A double layered lstm-rnn approach," in *Computer and Information Technology (ICCIT), 2017 20th International Conference of.* IEEE, 2017, pp. 1–6.

[12] M. M. H. Nahid, M. A. Islam, and M. S. Islam, "A noble approach for recognizing bangla real number automatically using cmu sphinx4," in *Informatics, Electronics and Vision (ICIEV), 2016 5th International Conference on.* IEEE, 2016, pp. 844–849.

[13] M. Alif Al Amin, M. Towhidul Islam, S. Kibria, and M. Shahidur Rahman, "Continuous bengali speech recognition based on deep neural network," 02 2019, pp. 1–6.

[14] J. Rahman Saurav, S. Amin, S. Kibria, and M. Shahidur Rahman, "Bangla speech recognition for voice search," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–4.

[15] A. Ahmed, Y. Hifny, K. Shaalan, and S. Toral, "Lexicon free arabic speech recognition recipe," in *International Conference on Advanced Intelligent Systems and Informatics.* Springer, 2016, pp. 147–159.

[16] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[17] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on.* IEEE, 2016, pp. 4960–4964.

[18] N. Jaitly, Q. V. Le, O. Vinyals, I. Sutskever, D. Sussillo, and S. Bengio, "An online sequence-to-sequence model using partial conditioning," in *Advances in Neural Information Processing Systems*, 2016, pp. 5067–5075.

[19] C. Raffel, M.-T. Luong, P. J. Liu, R. J. Weiss, and D. Eck, "Online and linear-time attention by enforcing monotonic alignments," *arXiv preprint arXiv:1704.00784*, 2017.

[20] Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. L. Y. Bengio, and A. Courville, "Towards end-to-end speech recognition with deep convolutional neural networks," *arXiv preprint arXiv:1701.02720*, 2017.

[21] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on.* IEEE, 2017, pp. 4845–4849.

[22] Y. Qian and P. C. Woodland, "Very deep convolutional neural networks for robust speech recognition," in *Spoken Language Technology Workshop (SLT), 2016 IEEE.* IEEE, 2016, pp. 481–488.

[23] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, 2016, pp. 173–182.

[24] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates *et al.*, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[25] S. H. Sumit, T. Al Muntasir, M. A. Zaman, R. N. Nandi, and T. Sourov, "Noise robust end-to-end speech recognition for bangla language," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP).* IEEE, 2018, pp. 1–5.

[26] A. Das and S. Bandyopadhyay, "Morphological stemming cluster identification for bangla," *Knowledge Sharing Event-1: Task*, vol. 3, 2010.

[27] M. N. Y. Ali, S. A. Al-Mamun, J. K. Das, and A. M. Nurannabi, "Morphological analysis of bangla words for universal

networking language," in *2008 Third International Conference on Digital Information Management*. IEEE, 2008, pp. 532–537.
[28] S. Sinha, A. Jain, and S. S. Agrawal, "Speech processing for hindi dialect recognition," in *Advances in Signal Processing and Intelligent Recognition Systems*. Springer, 2014, pp. 161–169.
[29] "Openslr - large bengali asr training data set," Google Inc., http://www.openslr.org/53/.
[30] M. Islam, "Research on bangla language processing in bangladesh: progress and challenges," in *8th International Language & Development Conference*, 2009, pp. 23–25.
[31] M. Hossain, M. Rahman, U. K. Prodhan, M. Khan *et al.*, "Implementation of back-propagation neural network for isolated bangla speech recognition," *arXiv preprint arXiv:1308.3785*, 2013.
[32] P. P. Shrishrimal, R. R. Deshmukh, and V. B. Waghmare, "Indian language speech database: A review," *International journal of Computer applications*, vol. 47, no. 5, pp. 17–21, 2012.
[33] M. A. Al Amin, M. T. Islam, S. Kibria, and M. S. Rahman, "Continuous bengali speech recognition based on deep neural network," in *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*. IEEE, 2019, pp. 1–6.
[34] E. D. J. F. B. G. M. H. A. J. M. E. P. M. J. R. A. R. S. P. W. S. R. S. E. T. J. W. Aric Bills, Anne David, "IARPA-babel103b-v0.4b LDC2016S08. Web Download. Philadelphia: Linguistic Data Consortium," https://catalog.ldc.upenn.edu/LDC2016S08, 2016.
[35] K. V. S. P. Rath, D. Povey and J. Cernocký, "Improved feature processing for deep neural networks," 2013, pp. 109–113.
[36] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition, in 'advances in neural information processing systems'," 2015.
[37] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *Seventh international conference on spoken language processing*, 2002.

Opinion Summarization of Bangla Texts using Cosine Simillarity Based Graph Ranking and Relevance Based Approach

Shofi Ullah, Sagar Hossain, K. M. Azharul Hasan
Department of Computer Science and Engineering
Khulna University of Engineering & Technology, Khulna-9203, Bangladesh

mail2safiullah@gmail.com, sagar13kuet@gmail.com, az@cse.kuet.ac.bd

***Abstract*— The main idea of the automatic extractive text or opinion summarization is to find most important representative small subset of the original document without any loss of important information. There are many existing methods available for text summarization of English, Turkish, Arabic and other languages. But very few attempts has been done for Bangla language because of its having rich morphology and multifaceted structure. In this paper, we propose a joint cosine simillarity based graph ranking and Relevance based scoring and ranking approach for the summarization of bangla text. We developed a stemming algorithm based on Parts of Speech(POS) tagging consisting of around two lakhs POS tags for Bangla texts. A redundancy removal algorithm is also proposed to remove redundancy so that each sentences in the summary represents exactly the most important information in the document. The performance of the proposed approach is evaluated by measuring the recall, precision and f-score based on Rouge metric and it is also showed that proposed approach outperforms to other existing summarization methods for Bangla texts.**

I. INTRODUCTION

The task of the text summarization is to create the shortest vertion of the original source texts preserving the most salient information. In this era of internet, it is the demand of the user to understand the huge amount of texts within a very short time. Nowadays online newspapers, websites and blogs in Bangla are increasing hugely. So, sometimes it is seen that articles headline is very attractive but found nothing informative reading the whole document. Therefore, a good precise summarization helps user to make decision whether to read the document or not by reading precise summary, also saves valuable time. Extractive and Abstractive are two types of summarization. Extractive summary creates summary with some representative sentences from original source texts where as abstractive creates summary with noble sentences not present in the original source texts. In this paper, we propose an extractive summarization approach for Bengali texts.

Different scoring and ranking algorithms were proposed based on features of query terms, sentence positional value [1], cue method, text title, term frequency [2], words position and key phrase frequency [3]. However, most of the cases promising performance could not be obtained. In this paper, we propose a joint cosine simillarity based graph ranking and relevance scoring based method which achieves very good precision and we showed our approach achieves better performance compared to others. Our main contributions towards the summarization for the bangla texts is summarized as follows:

- A stemming algorithm is proposed based on Parts of Speech(POS) tagging consisting of around 2 lakhs POS for Bangla Texts.
- Semantic similarity between the two sentences is measured through cosine simillarity measurement based approach using term frequency. Graph ranking algorithm is involved to rank the sentences based on their importance in the document.
- A relevance based scoring and ranking algorithm is proposed extracting the positions of the nouns in the sentences.
- A redundancy removal algorithm based on cosine simillarity is proposed to remove redundancy from sentences.

The rest of the paper is organized as follows; section II presents literature review, section III gives a description of our methodology, section IV analyzes the results of the experiments and section V concludes the paper by summarizing the findings.

II. RELATED WORK

Text summarization in Bangla has not been much successful due to several problems such as non-availability of digital standard text databases, inflectional morphology in Bangla. The first technique of automatic Bangla text summarization was proposed [5] using the features of query terms for document indexing and information retrieval. Based on features of location method, cue method, term frequency, word position and numerical data [3] [4], several research work Inspiring the summarization for the English document several research work have also conducted to Bangla texts. But most of the cases these approaches failed to achieve the promising performance as it is for being a different language.

978-1-7281-5242-4/19

Again extraction of key phrase based summarization [9] also had been proposed for Bangla texts based on key phrase frequency, term frequency and sentence position in the document. But some limitations were discussed[10] against this technique. Different sentence and graph scoring methods[6][7][8] were also proposed. Bangla text summarization using the features of part of speech (pos), title words, first paragraph words and words from last two sentences, key words retrieval[5][11] approaches were also proposed.

In recent years different machine learning and heuristic approaches has been proposed for bengali text summarization. Two text summarization aproaches proposed in [15] based on term frequency and semantic and word order simillarity of sentences in the content. Text summarization is also implemented through some set of heuristics rules [16] by analyzing text. K-Means clustering approach [17] is proposed based on tf-idf features and also considers cue and skeleton words for scoring the words. Finally a sentence score is obtained by summing up its all constituent words score.

However, all of the feature extraction based text summarizer did not ahieved promising performance towards the texts for Bangla. We developed a rich Bangla POS tagger consisting of around 2 lakhs words labelled with their POS. We have also shown that the proposed joint cosine simillarity based graph ranking and relevance based scoring and ranking method outperforms compared to other summarization systems in Bangla.

III. Proposed Work

From the source text document, first preprcessings are performed. Then introducing two salience scoring method, the score of each sentences is obtained and ranking is done in descending order of the score. Top scored n sentences are selected for summary from both approaches by union operation. To remove the redundancy of summary sentences we proposed a redundancy removal algorithm. In the next sections we described our proposed approaches ellaborately.

A. Preprocessing

Some preprocessing steps including tokenization, stop words removal and stemming with POS tagging are performed on the document. In the next subsections, we described about these preprocessing steps.

1) Tokenization: Sentence and word tokenizations are performed on the document. Sentences in the bengali texts are genarally ended with । or ? or ! character. Therefore sentences are extracted by seeing their appearences in the document. When all the sentences are extracted, redundant copies of a sentences are removed. Word tokenization separates words from the unstructured sentence and makes easy for performing operation on each words of the sentences. Therefore, a sentence is represented by a set of tokens or words.

2) Stop Words Removal: Stop words are the words which have less significance in texts and have very high or very low occurence in the document and they are removed. More than 300 stop words available for bengali texts shown some as follows:
অতএব, অথচ, অথবা, অন্তত, অন্য, অবশ্য, অর্থাত, আছে, আবার, আর, আরও, ইত্যাদি, এত, এতটাই

3) Stemming and Parts of Speech Tagging: Same words can be found in different lexicon order. Meaning of these words are mostly the same as their root word is same. By finding the proper root words the relationship between the sentences can be identified easily. A Bangla POS tagger proposed in [14] based on some rules and dictionary, stemmer and verb-dataset. We have developed a POS tagger consisting of around 2 lakhs words labelled with POS for different root words. To determine the stemming and POS tagging, the characters are first identified. If the word is in the POS, then labelling of word to POS POS(word) is done. Otherwise removing last character and matching the word without removed character with POS tags is done. But if there is a এ অথবা ই অথবা ঐ কার prefix infront of a letter in bangla word, then we removed those prefix এ, ই, ঐ কার first and then matched with POS list. If not matched with POS, then the postfix characters which residing after the letter are removed and again matched with POS tags. Again if not matched with the POS, then finally we removed the letter only and further checked with POS list for tagging the words. This process continues untill the word is labelled with a POS. There are infinitive number of nouns in the world. So, certainly when we checked POS of the such word which is not exist in our POS list, determination of the POS is not possible. Thats why at the end of algorithm when all the checking are done, the POS of the word is labelled as noun. The whole algorithm is illustrated as follows:

Algorithm 1: Stemming with POS tagging algorithm

Input: word
Output: word, POS(word)
temp ← *word*
while *len(word)* > 1 **do**
  if *word not in POS list* **then**
    i ← *length(word)*
    if *word[i] is* suffix কার *and word[i-2] is* এ or ই or ঐ (prefix কার) **then**
      remove word[i-2]
    else if *word[i-1] is* এ or ই or ঐ কার **then**
      remove word[i-i]
    else
      remove word[i]
  else
    return word, POS(word)
if *len(word)* < *2* **then**
  return temp, noun

B. Cosine Similarity Based Graph Ranking Algorithm

In the next sections we described ellaborately about the cosine simillarity measure, graph formation and graph ranking process.

1) Cosine Similarity Measure: Cosine similarity [12] is the cosine of the angle between two n-dimensional vectors in an n-dimensional space. It is the dot product of the two vectors divided by the product of the two vectors magnitudes. The cosine similarity between the two vectors A and B denoted by S is calculated using the following formula:

$$S = cos(\theta) = \frac{A.B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

To determine the cosine simillarity between the two sentences, they should be converted into vectors. Bag of words using term frequency(tf), term frequency-inverse document frequency(tf-idf) and word embeddings facilate the task of representing sentences as vectors. TF performs good for measuring the simillarity of texts compared to other approach. We used term frequency(tf) for measuring cosine simillarity between the sentences. The higher the value of $cos(\theta)$, the higher the simillarity of the two sentences. Calculation of the cosine simillarity for two sentences is shown as follows:

Sentence 1: রাহাত রোজ বিদ্যালয়ে যায় ।

Sentence 2: রাহাতের বিদ্যালয়ের শিক্ষকগণও রাহাতকে অনেক পছন্দ করেন ।

Step 1: We calculated the term frequency using Bag of Words for these two sentences:

TABLE I
VECTORIZATION USING TERM FREQUENCY

No	রাহাত	রোজ	বিদ্যালয়	যায়	শিক্ষক	পছন্দ	কর
S1	1	1	1	1	0	0	0
S2	2	1	0	0	1	1	1

Step 2: From following example it may realised that term-frequency favours the documents or sentences which are long. To solve this problem one way is to normalize the term frequencies with the respective magnitudes or L2 norms. L2 norm [13] is calculated as the square root of the sum of the squared vector values. Therefore, summing up squares of each frequency and taking a square root, L2 norm of Sentence 1 is 2 and Sentence 2 is 2.828. Now term-frequencies are divided by respective sentence L2 norm shown in table II.

TABLE II
NORMALIZATION OF TERM FREQUENCY

No	রাহাত	রোজ	বিদ্যালয়	যায়	শিক্ষক	পছন্দ	কর
S1	0.5	0.5	0.5	0.5	0	0	0
S2	0.707	0.35	0	0	0.35	0.35	0.35

Step 3: Now we calculated the cosine similarity with a dot product of S_1 and S_2 denoted as $Cosim(S_1, S_2)$ as we normalised two vectors.

$$Cosim(S_1, S_2) = 0.5 \times 0.707 + 0.5 \times 0.35 = 0.5285$$

By following these steps cosine simillarity of a sentence to all the sentences in the document is calculated and a two dimensional matrix is formed consisting the simillarity value of the sentences between each other.

2) Graph Formation: In this phase our target is to construct the graph with the sentences from the simillarity scores obtained from previous phase. Undirected weighted graph is build from the similarity matrix which is obtained from the previous phase. Each node in the graph represents a sentence. In order to add an edge between two nodes or sentences in the graph, we have identified the noun, pronoun, verb, adjective tokens from the sentences. An edge is established between the two nodes if there are at least one common tokens between the sentences.

3) Graph Ranking Algorithm: We employed a modified weighted graph based ranking algorithm[18], which will take into account the edge weights in the vertices in ranking process. The edge weight corresponds to cosine simillarity obtained from previous cosine simillarity matrix. We denoted the salience or importance score of predicate argument structure by $Pr(V_i)$. The importance score of a sentence can be achieved from all those sentences that are connected to it and formulated as follows:

$$Pr(V_i) = (1 - d_p) + d_p . \sum_{V_j \epsilon Adj(V_i)} \frac{Pr(V_j) \times W_{ji}}{\sum_{V_z \epsilon Out(V_j)} W_{jz}} \quad (2)$$

Here, d_p is the dumping factor, assigned value of 0.85 [18] and W is the cosine simillarity value between sentences. Initially all the score of the sentences is set to 0 for the optimal performance. The ranking algorithm keeps on computing the salience scores of the vertices untill the convergence is achieved. After that sentences are sorted in descending order based on their importance score. Now top n sentences having high scores are selected for summary sentences.

C. Relevance Based Approach

Nouns are the basis for the calculation of the sentence weight [19] [20]. In this phase, nouns are extracted from sentences and performed three operations on them. First distance among the nouns are calculated. Then relevance for each noun is measured and finally we obtained the score of a sentence by adding all the nouns relevance score in the sentence.

1) Distance Calculation of Nouns: In this step we calculated the distance between the nouns based on position of the nouns in the sentences. Distance between

the two nouns n_1 and n_2 is calculated using the following formula:

$$Distance(n_1, n_2) = \big|Position(n_1) - Position(n_2)\big|$$

2) Relevance Calculation: Relevance of a noun is calculated by summing the distance of all noun from the current one. Relevance of a noun is calculated using the formula as follows:

$$Rel(n_1) = \sum_{i=1}^{n} \big|Position(n_1) - Position(n_i)\big|$$

3) Relevance Calculation of Sentences: Now Relevance of a sentence is obtained by summing all the relevance score of nouns consisting the sentences. If there are m nouns in the sentences, then the relevance of a sentence(S) is calculated using the following formulas:

$$Rel(S) = \sum_{i=1}^{m} Rel(n_i)$$

The higher the relevance score of a sentence, the higher the sentence is important. Now all the sentences are sorted in descending order on their relevance score and top n sentences having higher scores are selected for summary.

D. Redundancy Removal Algorithm

Both from cosine simillarity based graph ranking and Relevance based approaches, top scored n sentences are selected for summary denoted by S using union operation. But it may exceeds from n sentences as we selected n sentences from both the approaches. Again it may happen that two sentence having semantically almost same, may come into summarization. So, to remove redundancy from S, first top scored sentence(S[1]) is selected in final summary. Now a sentence is taken from S which are not in final summary and checked the redundancy with the sentences in final summary one by one by cosine simillarity measure. If cosine simillarity exceeds the predefined threshold 0.875 [21] then the sentence is considered as redundant and not included in the summary, otherwise sentence is included in summary. This process continues untill the expected length of summary is met. We demonstrated our algoriithm as follows:

Algorithm 2: Redundancy Removal Algorithm

```
   Input: S, Expected Length, Threshold
   Output: Summary(Sum)
1  Sum ← S[1]
2  for i ← 2 to length(S)
     & length(Sum) < Expected Length do
3  |  Flag ← True
4  |  for i ← 1 to length(Sum) do
5  |  |  if cosim(Sum[j], S[i]) > Threshold then
6  |  |  |  Flag ← False
7  |  if Flag then
8  |  |  Sum ← Sum ∪ S[i]
9  |
```

IV. Experiment and Performance Analysis

In the next sections, descriptions of the dataset is described and performance of POS tagger and text summarizer are also analyzed.

A. Dataset

Dataset were collected from the survey held in Khulna University of Engineering and Technology (KUET) organized by Institutional Quality Assurance Cell (IQAC) based on a question **In what height would you like to see your beloved KUET by 2021 for the celebration of the 50 years of independence**. The collected opinions of the indivizuals consist in English, so we converted them to Bangla by Google Translator and also by hand and prepared the dataset.

B. Performance Evaluation and Result Analysis

Performance of the proposed approach for text summarization greatly depends on the performance of the POS tagger. For evaluating the performance of our proposed POS tagger, we collected a test data from prothom-alo online newspaper. We have tested around 9014 words to their POS. Now to calculate the performance we used the formula [14] as follows:

$$Accuracy = \frac{No\ of\ correctly\ detected\ POS\ tags}{Total\ words\ in\ the\ corpus}$$

We have correctly identified about 8043 words POS. In according to the above formula the accuracy of the proposed POS tagger is about 89.22%. Further improvement of the POS tagger is also continued. Two scoring and ranking method is involved to obtain the summary sentences. Now, to evaluate the performance of our proposed approach we compared the proposed method generated summary with the reference summary of the dataset provided. To evaluate the performance of the summary of the proposed method, we employed the Rouge [22] evaluation metrics. Rouge-1 works based on overlapping of the unigram of words and Rouge-2 works on the overlapping of bigrams of words between system generated or proposed summary with reference or dataset provided summary. Length for the summary is considered as 5% of the total length of the document as the reference summary have length of 5% of the document.

Total number of words in dataset provided or reference summary is denoted by R, proposed approach generated summary by P and total number of common overlapping words between reference and system generated summary by C, then recall, precision and f-score is defined as following:

$$Recall = \frac{C}{R}$$

$$Precision = \frac{C}{P}$$

$$F\text{-}score = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

The recall, precision and f-score is calculated by rouge-1 and rouge-2 metrics illustrated in the table III as follows:

TABLE III
PERFORMANCE EVALUATION BY ROUGE-1 AND ROUGE-2

Metric	Recall(%)	Precision(%)	F-score(%)
Rouge-1	83.38	76.92	80
Rouge-2	56.12	46.56	50.89

The performance of the proposed approach is compared against with semantic simillarity [15], K-Measn [16] and hueristic rule based [17] approaches by rouge-1 and rouge-2 metrics. In fig. 1 performance is compared througe rouge-1 metrics and it is clearly seen that the proposed approach f-score is better than the other three approaches. Again in fig. 2 comparison of performance score is shown for rouge-2 evaluation metric. Furthermore it is seen that performance score is improved for the proposed approach compared to other three approaches.

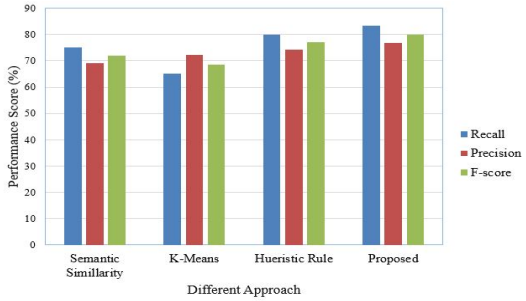


Fig. 1. Performace Score for Rouge-1

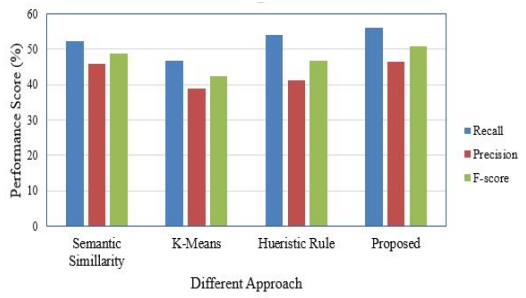


Fig. 2. Performace Score for Rouge-2

We conducted a survey among 10 students to give marks to proposed system summary aganist input dataset from 0 to 100 on their satisfaction. When our proposed method summary is about 5% of the dataset, then average satisfaction level is 80 and when it is 10% satisfaction level is increased to 96 which is really good for summarization. The generated summary of proposed approach for an original text taken from Prothom-alo is shown as follows:

Original Text: মাগুরা সদর উপজেলার কেচুয়াডুবি এলাকায় শ্যামলী পরিবহনের একটি বাস ও ট্রাকের মুখোমুখি সংঘর্ষে একজন নিহত হয়েছেন। আজ বৃহস্পতিবার ভোররাতের দিকে এ দুর্ঘটনা ঘটে। এ ঘটনায় ট্রাকের চালকসহ অন্তত ১০ জন আহত হয়েছেন। নিহত ব্যক্তির পরিচয় এখনো নিশ্চিত হওয়া যায়নি। পুলিশ ও আহত যাত্রীদের সঙ্গে কথা বলে জানা যায়, শ্যামলী পরিবহনের বাসটি ঢাকা থেকে সাতক্ষীরা যাচ্ছিল। আজ ভোররাত চারটার দিকে বিপরীত দিক থেকে আসা একটি ট্রাকের সঙ্গে বাসটির মুখোমুখি সংঘর্ষ হয়। এতে ট্রাকের সামনের অংশ দুমড়েমুচড়ে যায়। বাসের চালক নিয়ন্ত্রণ হারালে বাসটি রাস্তার পাশে একটি গাছের সঙ্গে ধাক্কা খায়। এতে ঘটনাস্থলেই নিহত হন ট্রাকের এক আরোহী। আহত ব্যক্তিদের মধ্যে ট্রাকের চালক আকতারুল (৪৫), ট্রাকচালকের সহকারী হাবিবুর রহমান, বাসচালকের সহকারী জুয়েলসহ বেশ কয়েকজন বর্তমানে মাগুরা ২৫০ শয্যা হাসপাতালে চিকিৎসা নিচ্ছেন। এর মধ্যে ট্রাকের চালক ও তাঁর সহকারীর অবস্থা গুরুতর। তাঁদের দুজনের বাড়িই চুয়াডাঙ্গা জেলায়। বাসচালকের সহকারী জুয়েল জানিয়েছেন, দুর্ঘটনায় বাসের ১০ থেকে ১২ জন আরোহী আহত হয়েছেন। তাঁদের যশোর ও মাগুরা হাসপাতালে ভর্তি করা হয়েছে। তাঁদের মধ্যে অনেকেই প্রাথমিক চিকিৎসা নিয়ে হাসপাতাল ছেড়েছেন।

Generated Summary: মাগুরা সদর উপজেলার কেচুয়াডুবি এলাকায় শ্যামলী পরিবহনের একটি বাস ও ট্রাকের মুখোমুখি সংঘর্ষে একজন নিহত হয়েছেন। আহত ব্যক্তিদের মধ্যে ট্রাকের চালক আকতারুল (৪৫), ট্রাকচালকের সহকারী হাবিবুর রহমান, বাসচালকের সহকারী জুয়েলসহ বেশ কয়েকজন বর্তমানে মাগুরা ২৫০ শয্যা হাসপাতালে চিকিৎসা নিচ্ছেন।

V. CONCLUSION

Our proposed approach mainly works on salience scoring and ranking. Top scored sentences are selected for summary and by redundancy removal algorithm redundancy from the selected summary sentences is removed. Our developed POS tagger for Bangla texts consisting of around 2 lakhs labelled POS words. Also further improvement on this POS tagger is also continueing as the Bangla language is very rich. Finally we showed our proposed techniques outperforms compared to other existing methods by focusing recall, precision and f-score.

REFERENCES

[1] K. Sarkar, "Bengali text summarization by sentence extraction" International Conference on Business and Information Management, NIT Durgapur, India, pp. 233-245, 2012.

[2] Md. Iftekharul Alam Efat, Mohammad Ibrahim, and Humayun Kayesh, "Automated Bangla Text Summarization by Sentence Scoring and Ranking" International Conference on Informatics, Electronics and Vision, Dhaka, Bangladesh, pp. 1-5, 2013.

[3] K. Sarkar, "An approach to summarizing Bengali news documents" International Conference on Advances in Computing, Communications and Informatics, Chennai, India, pp. 857-862, 2012.

[4] K. Sarkar "Sentence Clustering-based Summarization of Multiple Text Documents" International Journal of Computing Science and Communication Technologies, vol. 2, no. 1, pp. 325-335, 2009.

[5] Md Tawhidullslam and Shaikh Mostafa AI Masum, " Bhasa: A CorpusBased Information Retrieval and Summariser for Bengali Text" International Conference on Computer and Information Technology 2004.

[6] Md. Iftekharul Alam Efat, Mohammad Ibrahim and Humayun Kayesh, "Automated Bangla Text Summarization by Sentence Scoring and Ranking" International Conference on Informatics, Electronics & Vision (lCIEV), Dhaka, Bangladesh, pp. 1-5, 2013.

[7] Rafael Ferreira, Luciano de Souza Cabral, Rafael Dueire Lins, Gabriel Pereira e Silva, Fred Freitas, George D.C. Cavalcanti, Rinaldo Lima, Steven J. Simske, Luciano Favar, "Assessing sentence scoring techniques for extractive text summarization" Journal of Expert systems with applications, vol. 40, no. 14, pp. 5755-5764, 2013.

[8] Rejwanul Haque, Sudip Kumar Naskar, Andy Way, Marta R. Costa-jussa and Rafael E. Banchs, "Sentence similarity-based source context modelling in pbsmt" in Proceedings of the international conference on asian language processing, Harbin, China, pp. 257–260, 2010.

[9] Kamal Sarkar, "A Keyphrase-Based Approach to Text Summarization for English and Bengali Documents", International Journal of Technology Diffusion (IJTD), vol. 5, issue 2, pp. 28-38, 2014.

[10] Md Majharul Haque, Suraiya Pervinand and Zerina Begum, "Enhancement of keyphrase-based approach of automatic Bangla text summarization" IEEE Region 10 Conference (TENCON), Singapore, Singapore, pp. 42-46, 2016.

[11] Amitava Das and Sivaji Bandyopadhyay, "Topic-Based Bangla Opinion Summarization" International Conference in Social Computing, Beijing, China, pp. 675-682, 2010.

[12] Alfirna Rizqi Lahitani, Adhistya Erna Permanasari, Noor Akhmad Setiawan, "Cosine similarity to determine similarity measure: Study case in online essay assessment" 4th International Conference on Cyber and IT Service Management, Bandung, Indonesia, pp. 1-6, 2016.

[13] Sebahattin Bektas,Yasemin Sisman, "The comparison of L1 and L2-norm minimization methods" International Journal of Physical Sciences, vol. 5, no. 11, pp. 1721-1727, 2010.

[14] Md. Nesarul Hoque, Md. Hanif Seddiqui, "Bangla Parts-of-Speech Tagging using Bangla Stemmer and Rule based Analyzer" 18th International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, pp. 440-444 , 2015.

[15] Avik Sarkar, Md. Sharif Hossen, "Automatic Bangla Text Summarization Using Term Frequency and Semantic Similarity Approach" 21st International Conference of Computer and Information Technology (ICCIT), Dhaka, Bangladesh, pp. 1-6, 2018.

[16] Sheikh Abujar, Mahmudul Hasan, M.S.I Shahin, Syed Akhter Hossain, "A Heuristic Approach of Text Summarization for Bengali Documentation" 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Delhi, India, pp. 1-8, 2017.

[17] Sumya Akter, Aysa Siddika Asa, Md. Palash Uddin, Md. Delowar Hossain, Shikhor Kumer Roy and Masud Ibn Afjal, "An Extractive Text Summarization Technique for Bengali Document(s) using K-means Clustering Algorithm" International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Dhaka, Bangladesh, 2017.

[18] Atif Khanand Salim, Naomie Salim, Yogan Jaya Kumar, "Genetic semantic graph approach for multi-document abstractive summarization" International Conference on Digital Information Processing and Communications (ICDIPC), Sierre, Switzerland, pp. 173-181, 2015.

[19] Ahmad T. Al-Taani, Maha M. Al-Omour, "An extractive graph-based Arabic text summarization approach" The International Arab Conference on Information Technology, Jordan, 2014.

[20] Xiaojun Wan, Jianguo Xiao, "Single Document Keyphrase Extraction Using Neighborhood Knowledge" Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI, vol. 8, pp. 855-860, 2008.

[21] Faisal Rahutomo, Teruaki Kitasuka, Masayoshi Aritsugi, "Semantic Cosine Similarity" 7th International Student Conference on Advanced Science and Technology ICAST, 2012.

[22] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries" Text Summarization Branches Out: Proceedings of the ACL-04 Workshop, pp. 74-81, 2004.

A Bangla Text-to-Speech System using Deep Neural Networks

Rajan Saha Raju, Prithwiraj Bhattacharjee, Arif Ahmad, Mohammad Shahidur Rahman
Department of Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet, Bangladesh
{rajan10, prithwiraj12}@student.sust.edu, {arif_ahmad-cse, rahmanms}@sust.edu

***Abstract*—We present a Deep Neural Network (DNN) based statistical parametric Text-to-Speech (TTS) system for Bangla (also known as Bengali). A first step in building a DNN-based TTS system is having large speech data. Since good speech dataset for Bangla TTS is not available publicly, we created our own dataset for our system. We prepared a phonetically rich studio-quality speech database containing more than 40 hours of speech. The database consists of $12,500$ utterances. We also prepared a pronunciation dictionary (lexicon) of $1,35,000$ words for front-end text processing, which, to our knowledge, is the largest lexicon for Bangla. Our system extracts linguistic features from input text. Then it uses deep neural networks for mapping these linguistic features to acoustic features. We developed two TTS voices using our dataset - one male and one female voice. Both objective and subjective evaluation tests show that our system performs significantly better than the traditional Bangla TTS systems and is comparable to the commercially available best Bangla TTS system.**

***Index Terms*—spss, dnn, bangla speech corpus, lexicon, open source**

I. Introduction

Generating natural sounding speech from text (known as Text-to-Speech synthesis, or TTS) remains an interesting research problem despite decades of efforts. With the advent of smart devices, the necessity of TTS systems become more prevalent in recent days. TTS systems not only aid human-machine communication, but also help spreading knowledge and helping physically impaired people (e.g. blind people).

There are several ways to build a TTS system. Among them, two types of approaches dominant the development of TTS systems over the past few decades. One of the approaches, called concatenative synthesis [1] was the most popular method during the 1990s. Although this method produces highly natural synthetic voice, it requires a huge amount of human effort and expertise in specific areas. To build a typical concatenative unit-selection TTS, we need to record many hours of professional speech. Then we need to invest time in careful lexicon development and in creating complex rules for text normalization, among other things.

An attractive alternative of concatenative synthesis is the so called statistical parametric speech synthesis, or SPSS in short. The advantages of SPSS are the flexibility, control and small footprint, among others. In the SPSS systems, instead of concatenating smaller phonetic units, we generate acoustic parameters from which speech can be synthesized with the help of a vocoder. Among the earlier initiatives of developing SPSS systems, Hidden Markov Models (HMMs) based models [2] became common to the research community. A popular implementation of HMM-based approach is HTS [3], which was developed in the early 2000s. It has led the way in developing parametric synthesis approaches and algorithms. But due to the over-smoothing in acoustic modeling and the limitations of vocoders, HMM-based systems may not produce the most natural output [4].

The introduction of deep neural networks (DNNs) [5] opened a new research direction for acoustic modeling in SPSS [4] [6]. Although neural networks had been used in acoustic modeling in 1990s [7], it has shown impressive results [8] only in recent days thanks to the advancements in computational power and the availability of huge datasets. The research on SPSS systems gains a huge boost from that point on. The use of recurrent neural networks (RNNs) [4], long short-term memory RNNs (LSTM-RNNs) [9], and deep bidirectional LSTM-RNNs (BLSTM-RNNs) [10] improved the performance of SPSS systems significantly. SPSS systems thus become a strong contender of concatenative systems for producing natural sounding speech.

In spite of the advancements in the SPSS research, developing a good TTS system for an under-resourced language (like Bangla) still remains a challenging task. The lack of high quality speech data and the absence of language-specific text processing tools prevent us from building a high quality SPSS system. We intend to address these issues of Bangla TTS in this paper. The work we present here is based on the state-of-the-art algorithms of DNN-based synthesis. Our aim is to provide useful resources and recipes for developing Bangla TTS systems to the research community. Our contributions in this work include:

- We present a deep learning based Bangla Text-to-Speech system that can generate speech directly from Bangla text (in UTF-8 format) without any intermediate hand-engineered steps.
- We prepared 40 hours of studio-quality speech data for our TTS system. We employed two professional voice artists (one male and one female, both providing 20 hours of recordings each) to record the speech data. We plan

978-1-7281-5242-4/19

to release these speech data under liberal open source license to the research community.
- We developed a lexicon[1] of 1,35,000 words that can be used in a front-end text processing tool. The lexicon can also be used to develop an automatic grapheme-to-phoneme module for speech synthesis. We will also release this lexicon as an open source data.

The rest of the paper is arranged in the following order: we discuss the related research works in section II. After that, we describe our data preparation process in section III. The following section explains the model architecture in detail. Section V analyzes the performance of our system. Finally, we end the paper with some concluding remarks and by giving some directions to future improvements.

II. Related Works

Works on developing Bangla TTS systems are rare. The initial attempts on building Bangla TTS used concatenative approaches. Firoz Alam et. al. from BRAC University developed *Katha* [11] in 2007, which is a unit-selection TTS system based on Festival [12] toolkit. Abu Naser et. al. from Shahjalal University of Science and Technology developed a di-phone concatenation based TTS system *Subachan* [13] in 2009. An HMM-based SPSS system [14] was developed in 2012.

After the introduction of DNNs for acoustic modeling in SPSS systems, the TTS research has been accelerated by manyfolds. But we do not notice any significant efforts to develop DNN-based Bangla SPSS in any Bangladeshi or Indian institutions, although a commercial Bangla TTS was released by Google [15] in 2016. Google also released some TTS resources [16] to encourage Bangla research community to work on speech synthesis.

TTS researches can be benefitted by many open source tools developed and shared by prominent institutes and by individual researchers. Idlak Tangle [17] is one such TTS system which is developed based on another open source speech recognition system Kaldi [18]. Merlin [19] is another open source DNN-based TTS system developed in the University of Edinburgh. The developers of Merlin uses Ossian [20] and Festival for front-end text processing, and the WORLD [21] vocoder for synthesizing speech from acoustic parameters. All of those tools are freely available under open source licenses. These resources are now being used to develop TTS systems for under-resourced languages, such as [22].

Although SPSS methods were originally preferred over concatenative methods due to their small footprint, they still require a huge amount of human intervention specially for language-specific text processing. To resolve this issue, modern researchers are trying to develop end-to-end TTS systems where speech is generated directly from (text, speech) pairs without any intermediate steps. Many of the giant tech companies have been engaged in this research. They have been getting promising results in the last couple of years. Example of such attempts are WaveNet [23], Tacotron [24], and Tacotron 2 [25] from Google, Deep Voice 1 [26], 2 [27], 3 [28] from Baidu, Char2Wav [29] from MILA, FastSpeech [30] from Microsoft, etc. Although the commercial companies do not share their code publicly, many individual implementations of these systems are available online which are almost as good as the original ones. New TTS researchers can be benefitted from these open source implementations.

We employed some good implementations of open source tools for our Bangla SPSS system, which will be discussed in section IV. The following section describes how we prepare our speech data.

[1]A lexicon is a dictionary of pronunciation.

III. Data Preparation

The first step of building a parametric voice is to collect a large amount of speech data, along with the associated transcriptions. There is no good quality public TTS data available for Bangla. Google has released their Bangla TTS data [16], but it contains only three hours of speech recorded with multiple speakers. A few more public dataset are available, all of which contains merely a couple of hours of speech. Those datasets were prepared for unit-selection TTS and for acoustical analysis of Bangla speech. So we needed to prepare our dataset from scratch.

At first, we consulted the literature for developing *phonetically balanced* text corpus [31]–[33]. Then we gather text data from various domains. We ensured that our text corpus contains all possible pronunciations of Bangla. Our final corpus contains more than 12,000 utterances, each consisting of stand-alone sentences. A summary of our dataset is presented in table I.

TABLE I
Summary of speech data prepared for Bangla SPSS

Total sentences	12,537
Total words	1,22,627
Total unique words	24,582
Minimum words in a sentence	3
Maximum words in a sentence	20
Average words in a sentence	9.78
Total duration of speech (hours)	20 : 14 : 21
Average duration of each sentence (seconds)	5.81

After preparing the text corpus, we started recording the speech. We have built a sound-proof audio recording studio solely for this purpose. Then we employed two professional voice artists (one male and one female), both of whom have a clear, strong voice and can record speech for several consecutive hours. We recorded their voices in the raw wave format, at a sampling rate of 48 KHz. The total duration of collected speech is around 20 hours (for each speaker).

IV. Model Architecture

Figure 1 shows the steps in building the Bangla SPPS system. The major processing units of this system are: (i) a text normalizer, (ii) a front-end for processing text input, (iii) & (iv) two deep neural networks (DNNs) for duration and acoustic modeling, and (v) a vocoder for synthesizing speech

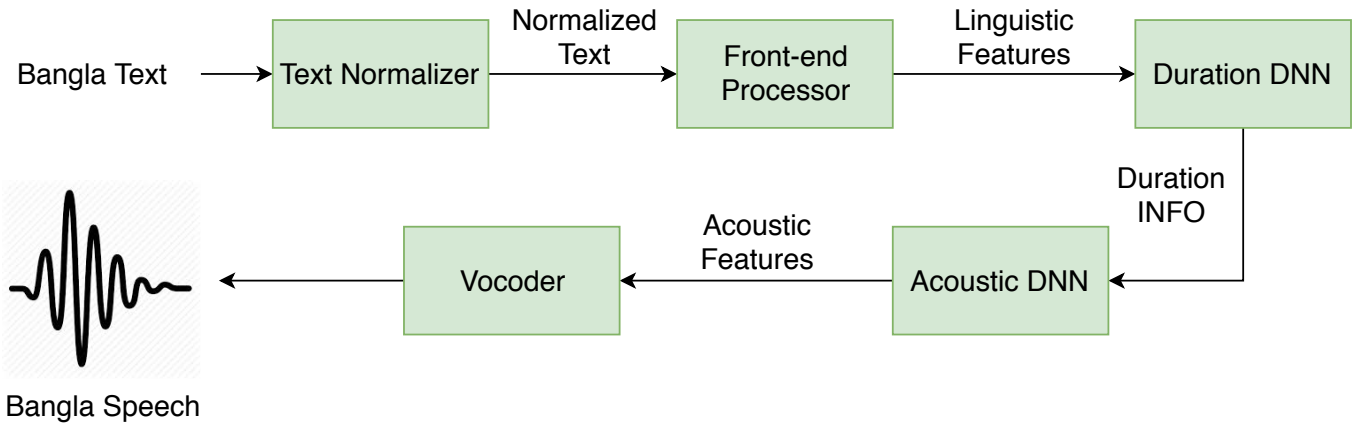


Fig. 1. Architecture of Bangla SPSS

from acoustic parameters. An elaborate description of these units are given below.

A. Text Normalizer

Text normalization is the process that converts non-standard words (NSWs) into pronunceable forms. In a typical TTS system, the raw input text is passed to a text normalizer, and a *normalized* text is produced as output. We developed a text normalizer for our Bangla SPSS system. To develop the normalizer, we consulted the literature thoroughly and implemented some methods proposed by the researchers [34].

B. Front-end

A DNN-based TTS systems need a front-end text processor to get the linguistic features from the input text. We have successfully utilized two open source front-end tools - Ossian [20] and Festival [12].

The Ossian open-source text processor allows us to extract linguistic features from the input text. It is language-independent, and can work with any text given in the UTF-8 format. It performs the feature extraction by mapping each character of the input text to a corresponding phoneme. Although modeling speech units from characters is a naive technique, it works surprisingly well if provided with a sufficiently large dataset and a phonetically balanced text corpus.

To obtain a better control over the linguistic feature computations, we might want to use a language-specific front-end tool. The Festival speech synthesis tool can help us in this regard. The front-end text processor of festival is language specific. It requires a well-defined lexicon and phonology (collection of phonetic information) of the target language. We have prepared a lexicon of 1,35,000 words and used a grapheme-to-phoneme converter [35] to handle out-of-vocabulary words. We have adopted and updated a phonology released by Google [16].

The front-end outputs HTS-style [36] labels with the state-level alignment. From these labels, a vector of linguistic features are generated by Ossian (or Festival). This feature vector is then fed to the duration model and acoustic model.

C. Duration Model

The back-end of our Bangla TTS system consists of two deep neural networks. The first one, duration DNN, takes linguistic features generated from front-end processing as input, and learns the proper duration information by updating weights. We split the training data into three sets: training (94%), testing (3%) and validation (3%).

We employed feed forward networks for both duration and acoustic modeling. The Merlin toolkit allows us to use other variants of neural networks as well. The basic architecture of the preferred network can be specified in a configuration file and the model will perform accordingly. We get this flexibility due to the availability of recipe-like structure implemented by the Merlin developers.

Our duration DNN consists of 3 layers of hidden units where each layer contains 512 neurons. The network uses Gradient Descent optimizer with the learning rate of 0.002. The output of this network is then fed to the acoustic model.

D. Acoustic Model

The acoustic DNN was trained to map the input linguistic features and the associated duration features into acoustic features. Linguistic features (sequence binary vectors) are normalized in the range of [0.01, 0.99] before passing to input layers of DNN. The acoustic DNN consists of 6 layers of hidden units where each layer contains 1024 neurons. The network applies the Gradient Descent algorithm with learning rate 0.002 to minimize the errors between predicted outputs and target outputs and updates the weights in every iteration.

E. Vocoder

Acoustic features, the output of acoustic modeling DNN are normalized appropriately so that they can be used by a vocoder. They are reduced to zero mean and unit variance. Finally, acoustic features are sent to a vocoder for synthesizing waveform. We used WORLD [21], an open source vocoder modified by Merlin for making compatible with the entire system.

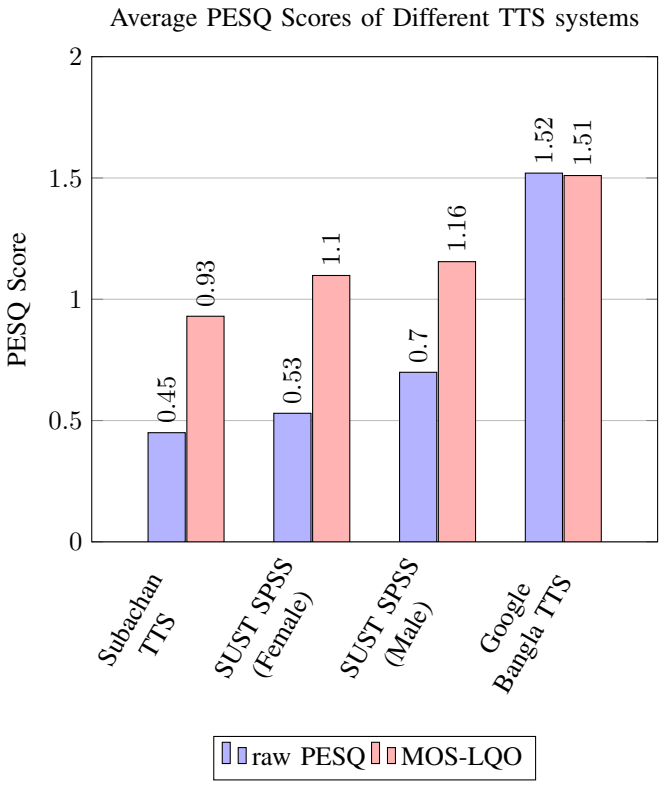
V. Results and Discussion

To assess the quality of our proposed system, we conducted both objective and subjective evaluation.

A. Objective Evaluation

We choose the Perceptual Evaluation of Speech Quality (PESQ) [37] score, specifically raw-PESQ and MOS-LQO for the purpose of objective evaluation. The intervals of raw-PESQ and MOS-LQO are [-0.5, 4.5] and [1, 5], respectively. In the measurement of PESQ, two waveforms (one is original and the other is synthetic) are required. The experimental set-up for determining the PESQ score is as follows.

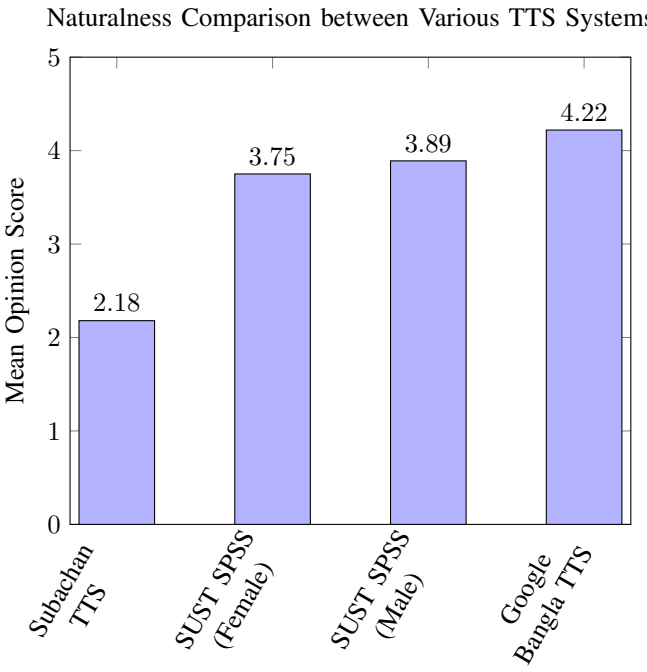
We calculated the PESQ scores of 4 systems (Subachan TTS, SUST SPSS Female, SUST SPSS Male, Google Bangla TTS) to compare speech quality. For calculating the PESQ scores of the TTS systems, we picked 100 random sentences and corresponding speech recordings as original speech data. Then, we synthesized 100 waveforms of selected sentences from the TTS systems as synthetic speech. Finally, the original and synthetic speech was sent to the PESQ system in pairs for determining the PESQ score. The following bar chart shows the average PESQ scores for the four systems mentioned above.



B. Subjective Evaluation

A benchmarking test for the TTS system is the Mean Opinion Score (MOS). So we selected the MOS test for our subjective evaluation. We invited some native Bangladeshi speakers to test the naturalness of various systems. 30 people volunteered us as the listeners for the MOS test. The listeners are aged between 20 and 35. 18 of them were male, rest were female. The listeners listened to 20 synthetic sentences generated by various TTS systems and gave a naturalness score between 0 and 5 to each of the systems. A higher score means better naturalness. All the scores are averaged to obtain the *mean score* of a system. In fact, we excluded the highest and lowest score obtained by each system to calculate a more accurate MOS named Robust-MOS.

The following bar chart summarizes the MOS scores obtained for various systems. The results of our MOS test supports our objective evaluation tests as well. Our system clearly outperforms the publicly available concatenative TTS system *Subachan*. Our male voice performs slightly better than the female voice. Both of the voices are comparable with the best known commercial Bangla TTS from Google.



VI. Conclusion

In this paper, we aim to address some of the challenges in the area of Bangla SPSS. We developed a large speech corpus and efficiently trained a deep neural network to synthesize Bangla speech. We also developed a large lexicon that will help researchers build a robust TTS front-end. Experimental analysis shows that our system performs significantly better than the traditional TTS systems, in terms of naturalness. Despite the promising performance, we still need to work harder in improving the scalability of the system, linguistic components, lexicon and text normalization. We hope that, with the availability of resources we developed, our research community will come forward to contribute more to Bangla speech synthesis research.

References

[1] A. J. Hunt and A. W. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, vol. 1. IEEE, 1996, pp. 373–376.

[2] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Simultaneous modeling of spectrum, pitch and duration in hmm-based speech synthesis," in *Sixth European Conference on Speech Communication and Technology*, 1999.

[3] H. Zen, T. Nose, J. Yamagishi, S. Sako, T. Masuko, A. W. Black, and K. Tokuda, "The hmm-based speech synthesis system (hts) version 2.0." in *SSW*. Citeseer, 2007, pp. 294–299.

[4] H. Ze, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *2013 ieee international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 7962–7966.

[5] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, 2012.

[6] Z.-H. Ling, S.-Y. Kang, H. Zen, A. Senior, M. Schuster, X.-J. Qian, H. M. Meng, and L. Deng, "Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 35–52, 2015.

[7] T. Weijters and J. Thole, "Speech synthesis with artificial neural networks," in *IEEE International Conference on Neural Networks*. IEEE, 1993, pp. 1764–1769.

[8] O. Watts, G. E. Henter, T. Merritt, Z. Wu, and S. King, "From hmms to dnns: where do the improvements come from?" in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5505–5509.

[9] Y. Qian, Y. Fan, W. Hu, and F. K. Soong, "On the training aspects of deep neural network (dnn) for parametric tts synthesis," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 3829–3833.

[10] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, "Tts synthesis with bidirectional lstm based recurrent neural networks," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[11] F. Alam, P. K. Nath, and M. Khan, "Text-to-speech for bangla language using festival," 2007.

[12] "The festival speech synthesis system," accessed: 2019-07-28. [Online]. Available: http://www.cstr.ed.ac.uk/projects/festival/

[13] A. Naser, D. Aich, and M. R. Amin, "Implementation of subachan: Bengali text-to-speech synthesis software," in *International Conference on Electrical & Computer Engineering (ICECE 2010)*. IEEE, 2010, pp. 574–577.

[14] S. Mukherjee and S. K. D. Mandal, "A bengali hmm based speech synthesis system," *arXiv preprint arXiv:1406.3915*, 2014.

[15] A. Gutkin, L. Ha, M. Jansche, K. Pipatsrisawat, and R. Sproat, "Tts for low resource languages: A bangla synthesizer," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, 2016, pp. 2005–2010.

[16] "Google international language resources," accessed: 2019-07-28. [Online]. Available: https://github.com/google/language-resources/blob/master/bn/festvox/phonology.json

[17] B. Potard, M. P. Aylett, D. A. Baude, and P. Motlicek, "Idlak tangle: An open source kaldi based parametric speech synthesiser based on dnn." in *INTERSPEECH*, 2016, pp. 2293–2297.

[18] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[19] Z. Wu, O. Watts, and S. King, "Merlin: An open source neural network speech synthesis system." in *SSW*, 2016, pp. 202–207.

[20] "Ossian: A simple language independent text to speech front-end," accessed: 2019-07-28. [Online]. Available: https://github.com/CSTR-Edinburgh/Ossian

[21] M. Morise, F. Yokomori, and K. Ozawa, "World: a vocoder-based high-quality speech synthesis system for real-time applications," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[22] A. Deka, P. Sarmah, K. Samudravijaya, and S. Prasanna, "Development of assamese text-to-speech system using deep neural network," in *2019 National Conference on Communications (NCC)*. IEEE, 2019, pp. 1–5.

[23] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[24] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio *et al.*, "Tacotron: Towards end-to-end speech synthesis," *arXiv preprint arXiv:1703.10135*, 2017.

[25] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4779–4783.

[26] S. Ö. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, A. Ng, J. Raiman *et al.*, "Deep voice: Real-time neural text-to-speech," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 195–204.

[27] A. Gibiansky, S. Arik, G. Diamos, J. Miller, K. Peng, W. Ping, J. Raiman, and Y. Zhou, "Deep voice 2: Multi-speaker neural text-to-speech," in *Advances in neural information processing systems*, 2017, pp. 2962–2970.

[28] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller, "Deep voice 3: Scaling text-to-speech with convolutional sequence learning," *arXiv preprint arXiv:1710.07654*, 2017.

[29] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. Courville, and Y. Bengio, "Char2wav: End-to-end speech synthesis," 2017.

[30] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech: Fast, robust and controllable text to speech," *arXiv preprint arXiv:1905.09263*, 2019.

[31] P.-E. Honnet, A. Lazaridis, P. N. Garner, and J. Yamagishi, "The siwis french speech synthesis database? design and recording of a high quality french database for speech synthesis," Idiap, Tech. Rep., 2017.

[32] R. Sonobe, S. Takamichi, and H. Saruwatari, "Jsut corpus: free large-scale japanese speech corpus for end-to-end speech synthesis," *arXiv preprint arXiv:1711.00354*, 2017.

[33] L. Gabdrakhmanov, R. Garaev, and E. Razinkov, "Ruslan: Russian spoken language corpus for speech synthesis," *arXiv preprint arXiv:1906.11645*, 2019.

[34] M. M. Rashid, M. A. Hussain, and M. S. Rahman, "Text normalization and diphone preparation for bangla speech synthesis." *Journal of Multimedia*, vol. 5, no. 6, pp. 551–559, 2010.

[35] A. Ahmad, M. Raihan Hussain, M. Reza Selim, M. Zafar Iqbal, and M. Shahidur Rahman, "A sequence-to-sequence pronunciation model for bangla speech synthesis," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sep. 2018, pp. 1–4.

[36] H. Group *et al.*, "Hmm/dnn-based speech synthesis system (hts)," *Available in: http://hts. sp. nitech. ac. jp*.

[37] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, vol. 2. IEEE, 2001, pp. 749–752.

Data Set For Sentiment Analysis On Bengali News Comments And Its Baseline Evaluation

Md. Akhter-Uz-Zaman Ashik
Department of CSE
Shahjalal University of Science and Technology,
Sylhet,Bangladesh
Email:ashikchowdhury76@gmail.com

Shahriar Shovon
Department of CSE
Shahjalal University of Science and Technology,
Sylhet, Bangladesh
Email: shahriarshovon69@gmail.com

Summit Haque
Department of CSE
Shahjalal University of Science and Technology,
Sylhet, Bangladesh
Email: summit.haque@gmail.com

***Abstract*—The biggest challenge of Bengali language processing is creating a strong data set to do research on. The main focus of this paper is to introduce an authentic and credible data set[1] for Bengali sentiment analysis where the data was extracted from a well known online news portal's user comments. Here comments on various news were scraped, and for detecting the true sentiments of the sentences, five labels of sentiments were used. An online crowd sourcing platform was used for data annotation. To ensure the credibility and validity of the data set, every entry of the data set was tagged three times. Three models of text classification were used for baseline evaluation to check the validity of the data set. This data set might be of valuable help for future works and researches on Bengali sentiment analysis.**

Keywords:: **Sentiment Analysis, Data Set, Bengali, News Comments, SVM, RNN, LSTM, CNN.**

I. Introduction

Peoples value and opinion have a strong impact in today's world. In a world where almost everything is online, we get a large amount of feedback from people in these online sites, blogs and forums. Sentiment Analysis is the method to get a comprehension of this huge amount of opinion from people in a technological way. SA(sentiment analysis) has revolutionized the way we perceive data and make a decision out of this growing amount of data.

People like to express their opinion in online sites, and preferably in their own language. Bengali is a very widely spoken language. Many Bengali news portals have gained popularity in recent decade. SA is fairly very new to Bengali language as opposed to English.

A. Motivation

Language is often vague or highly contextual which makes it very difficult for a machine to understand without human help. As such, human annotated data is essential when training a machine learning platform to analyze sentiment. The performance of a machine learning model to detect sentiments depends largely on the training data. For this reason three different perspectives have been used to make the data set more precise.

There are plenty of scopes to do analysis and research on the aspect of sentiment analysis on Bengali language. Countless news portals, social sites, blogs, etc. have been on the rise which are using this language. These online sites can throw valuable insight on the peoples sentiment as a whole. One of the major challenge of performing sentiment analysis on Bengali language is creating a authenticated and credible data set without ambiguous data. If the data is classified under the opposite category by two different person while preparing the data set, then we can call it 'ambiguous'. The data set on which the sentiment analysis is to be done has to be free of these sort of data. One other thing that inspired us was when a data is labeled according to a persons opinion, that person's opinion is not judged whether s/he is right or not. So we decided to create a data set where the data set will not be labeled according to just one person's opinion to ensure the credibility of the data set.

II. Related Work

Sentiment Analysis means the characterization of the sentiment content of a text unit using Natural Language Processing, statistics or Machine Learning methods. The works of Minqing Hu and Bing Liu circa 2004 done on customer reviews was the first major work done on sentiment analysis. They proposed the Feature-Based Opinion Mining Model which is now known as Aspect-Based Opinion Mining[1]. Md. Atikur Rahman and Emon Kumar Dey presented a work based on their data sets for ABSA(aspect based sentiment analysis)[2]. Their data set had around 5092 data in total. They worked with three tags positive, negative, neutral. M. Nabil et al created a data set from Arabic sentiment tweets consisting of 10,000 tweets and did experiments with 4 way sentiment classification[3]. Akhtar, Md Shad et al created a data set for

[1]Sample of data set:https://github.com/Shahriar666/Sample_Data

978-1-7281-5242-4/19

aspect based sentiment analysis in Hindi and did it's baseline evaluation for data validation where the data set contains 5,417 review sentences across 12 domains. There are a total of 2,290 positive, 712 negative, 2,226 neutral and 189 conflict reviews [4]. A. K. Paul and P. C. Shill did sentiment analysis using mutual information for feature selection and multinominal Naive Bayes for classification using English language data, they have achieved 85.1% accuracy without using negation and got 85.8% accuracy with negation using English testing data. For Bengali, using Bengali testing data, they got 84.78% accuracy without using negation and got 83.77% accuracy with negation[5]. Sentiment analysis was also done in micro blog posts by S. Chowdhury and W. Chowdhury. They used support vector machine and maximum entropy to do the comparative analysis of these two machine learning algorithms[6]. M. S. Islam et al presented a research paper where six different approaches were discussed to evaluate the sentiment. They also discussed about the implementation of cosine similarity using TF-IDF to determine sentiments more accurately[7].

III. Methodologies

A. Data Collection

Online news papers have a huge collection of user comments. The data collected was from a widely popular online news portal Prothom-Alo's user comments [2]. From this huge collection of comments 10 specific fields were selected. There are other fields where users comment expressing their opinion but compared to these 10 fields they are negligible. These 10 fields were selected as these are fairly common in comments.

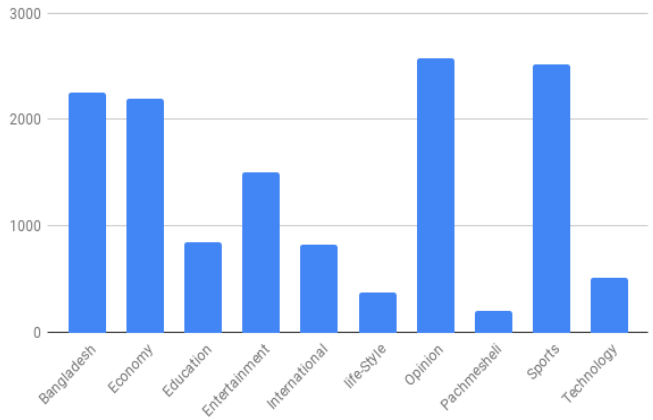


Fig. 1. Categories of the comments

Priorities were given to "Bangladesh", "Economy", "Opinion" and "Sports" because of being the most occurring theme among the comments.

[2]prothomaalo.com

B. Data Pre-processing

The data which was crawled had some characteristic problems within it. The issues we faced were:

- Sentences got divided and separated while crawling
- Some sentences were not properly structured
- Same sentences appeared more than once
- Some sentences had signs and unnecessary characters

To solve these problems the sentence which was not properly structured was dropped. Some signs like multiple question marks, multiple dots and exclamatory signs were cleared. The comments which appeared more than once were dropped.

C. Data Annotation

We wanted to create our own data set with proper attention given to the credibility of the sentiment behind the sentence. When a sentence is labelled, a single tag cannot ensure the actual sentiment of the sentence. A sentence might seem negative to an individual but might not be for another individual. The data set has each entry tagged by three different individuals to get three different perspectives.

There are many approaches to data labelling. Among them, crowd-sourcing is a convenient approach for sentiment analysis. Crowdsourcing is a practice in which information or inputs are obtained from a huge number of people, typically via the Internet. We used this practice for data annotation. Pipilika's crowdsourcing platform [3] was used for campaigns. The data set consists of the standard 5 category sentiments which are strongly positive, positive, neutral, negative, strongly negative where every sentence is labelled 3 times by 3 different individuals to ensure credibility.

D. Processing

Every entry having been tagged three times, they had to be processed and turned into one single tag. So we decided to assign a particular value against every tag. Then by calculating these values we decided to choose the final tag. Suppose an entry has three tags, one positive and two negative, as negative tag appears the most, that entry was labelled to be negative. There were some data where the data was tagged with three different labels which makes the data ambiguous. So we decided to drop entries like this. As a result the data set is free from ambiguous data.

E. Data Set Statistics

The data set we built has 13809 entries in it. It has mainly 5 Labels of Sentiment:

[3]crowd.pipilika.com

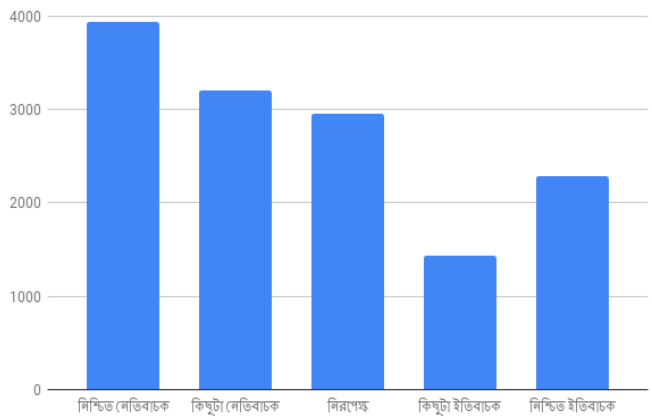


Fig. 2. Labels and there amount

TABLE I
LABEL TYPE AND COUNT

Label	Count
Slightly Positive	1436
Positive	2279
Neutral	2955
Negative	3936
Slightly Negative	3203

An overall map of the percentage of the sentiments is given below:

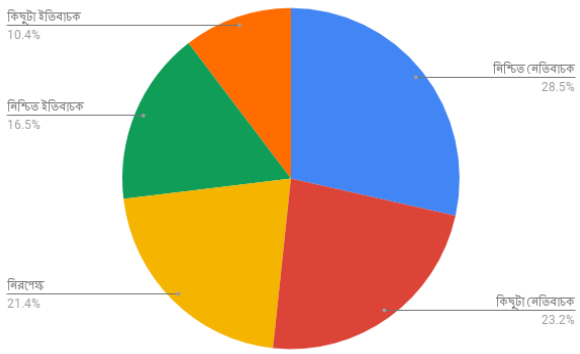


Fig. 3. Percentage of each labels

The data set does not have a negative or positive bias. The conventional model evaluation methods do not accurately measure model performance when faced with imbalanced data-sets. Standard classifier algorithms like Decision Tree and Logistic Regression have a bias towards classes which have number of instances. They tend to only predict the majority class data. The features of the minority class are treated as noise and are often ignored. Thus, there is a high probability of wrong classification of the minority class as compared to the majority class[8].
The statistical summary of the data set:

TABLE II
DATA SET STATISTICS

Category	Words
Total	248562
Longest Sentence	118
Average Sentence Length	44
Numeric Words	2389
Bengali Words	244432
Non-Bengali Words	4130

The various topics we tried to cover are given below in the table:

TABLE III
TOPICS OF THE DATA SET

Fields of Data	Number
Opinion	248562
Sports	118
Bangladesh	44
Economy	2389
Entertainment	244432
International	4130
Education	4130
Technology	4130
Lifestyle	4130
Various	4130

Opinion, Sports, Bangladesh and Economy are the majority of the topics covered.

IV. BASELINE EVALUATION

There are mainly three ways for classifying the sentiment of a text unit[9]. These are Machine Learning Techniques, Lexicon-Based Techniquesand and Hybrid Techniques.

Sentiment Analysis uses the evaluation metrics of Precision, Recall, F-score, and Accuracy for the classification problem. Also, average measures like macro, micro, and weighted F1-scores are useful for multi-class problems. Based on the balance of classes of the data-set the appropriate metric should be chosen. Four effective measures have been selected for the study based on the confusion matrix of the output. These are:

$$Precision(P) = TP/(TP + FP) \quad (1)$$

$$Recall(R) = TP/(TP + FN) \quad (2)$$

$$Accuracy(A) = (TP+TN)/(TP+TN+FP+FN) \quad (3)$$

$$F1 - Score(F_1) = 2.(P.R)/(P + R) \quad (4)$$

Our focus point here is performing Baseline Evaluation on the data-set that we have created on the Bengali news comments for sentiment analysis, where we have selected three models for the Baseline Evaluation. These are: Binary SVM classifier, Multi-class SVM classifier and LSTM(Neural Network).

A. SVM classifier

In machine learning, support-vector machines are supervised learning models that have associated learning algorithms which analyze data used for classification and regression analysis[10]. We can use an SVM classifier when the data has exactly two classes. An SVM classifies data by finding the best hyperplane that separates all the data points of one class from those of the other class.

The data-set was split into test data and train data in the following way:

- Amount of Train set: 10809
- Amount of Test set: 3000
- Amount of Validation Set: 2809

SVM classifier is pretty good for binary classification, when the number of categories is only two, negative and positive. Taking the *slightly positive* and *positive* classes into the *positive* label and the rest into the *negative* label, and dropping the *neutral* class

Following is the confusion matrix we have generated using the SVM classifier:

TABLE IV
CONFUSION MATRIX OF BINARY SVM CLASSIFIER

	Predicted Positive	**Predicted Negative**
Actually Positive	2823	892
Actually Negative	1714	5425

The following table shows the overall accuracy:

TABLE V
ACCURACY OF BINARY SVM CLASSIFIER

Model	**Test(%)**	**Train(%)**	**Validation(%)**
Binary SVM classifier	66.232	93.397	67.488

If we take the *Precison*, *Recall*, *Accuracy* and *F1-Score* into consideration of this model, we generate the table:

TABLE VI
EVALUATION OF THE SVM MODEL

Precision	**Recall**	**Accuracy**	**F1-score**
57.59	72.41	61.34	63.97

B. RNN(LSTM)

Recurrent Neural Networks(RNN) and Long Short-Term Memory(LSTM) networks are often used for sentiment analysis. Recurrent Neural Networks and Long Short-Term networks introduces a memory into the model. Having a memory in a network is useful because, when dealing with text data, the meaning of a word depends on the context of the previous text. A drawback of the Recurrent Neural network is that it is only capable of dealing with short-term dependencies. Long Short-Term Memory networks address this problem by introducing a long-term memory into the network[11]

We created the training , testing and validation splits as follows:

- Amount of Train set: 11041
- Amount of Test set: 1380
- Amount of Validation set: 1381

Like the SVM classifier, we took the *slightly positive* and *positive* classes into the *positive* label and the rest into the *negative* label, and dropping the *neutral* class, we did the evaluation.

The confusion matrix came up to be the following:

TABLE VII
CONFUSION MATRIX OF LSTM CLASSIFIER

	Predicted Positive	**Predicted Negative**
Actually Positive	2529	1151
Actually Negative	1495	5867

The following table shows the accuracy of this model:

TABLE VIII
ACCURACY OF THE LSTM MODEL

Model	**Test(%)**	**Train(%)**	**Validation(%)**
LSTM	74.741	96.967	78.833

By using the measurements of *Precison*, *Recall*, *Accuracy* and *F1-Score*, we can generate the table:

TABLE IX
EVALUATION OF THE LSTM MODEL

Precision	**Recall**	**Accuracy**	**F1-score**
72.84	87.22	74.74	79.291

C. CNN

Convolutional Neural Networks(CNN) is a class of Deep Neural Networks. CNNs are multi-layer perceptrons that are regularized. Multi-layer perceptrons usually have fully connected networks, which can cause over fitting. CNNs solve this issue by taking advantage of hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns.

For text classification with CNN, we usually embed the words of a sentence into a *2D* array stacking them together. Convolution filters are applied to selective number of words to produce a new feature representation. Then some pooling is performed on new features, and the pooled features from different filters are concatenated with each other to form the hidden representation. These representations are then followed by one (or multiple) fully connected layer(s) to make the final prediction[12].

We split the data in the following way for the evaluation:

- Amount of Train Data: 10809
- Amount of Test Data: 3000
- Amount of Validation Data: 2809

The confusion matrix for CNN:

TABLE X
CONFUSION MATRIX OF CNN CLASSIFIER

	Predicted Positive	**Predicted Negative**
Actually Positive	2144	1373
Actually Negative	1862	5455

The accuracy table for CNN:

TABLE XI
ACCURACY OF THE CNN MODEL

Model	**Test(%)**	**Train(%)**	**Validation(%)**
CNN	60.49	89.03	63.68

The evaluation of the CNN model:

TABLE XII
EVALUATION OF THE CNN MODEL

Precision	**Recall**	**Accuracy**	**F1-score**
58.92	68.52	60.49	66.24

V. DISCUSSION

SVM employs kernel tricks and maximal margin concepts to perform better in non-linear and high-dimensional tasks. SVMs are great for relatively small data sets with fewer outliers.

Neural networks typically perform better on very large data-sets. Neural networks profit a lot if the data points are structured in a way that can be exploited by the architecture. That is the case with our data-set, thats why neural networks (LSTM) is a better choice. Neural Networks may require more data but they almost always come up with a pretty robust model.

Deep learning really shines when it comes to complex problems such as image classification, natural language processing, and speech recognition. The data set has no clear distinct pattern which can be exploited by the CNN model.

VI. CONCLUSION

In our endeavour, we have created our very own data set for analysis, which consists of 13809 entries from the Prothom-Alo news portal. Baseline model selection and evaluation have been done on this data set using three different models which are SVM, CNN and LSTM model. We have done some comparative performance testing with our data set related to Bengali sentiment analysis and presented them in a tabular form.

We hope that this Data Set will be helpful for other researchers in the field of Natural Language Processing and Sentiment Analysis in Bengali language.

ACKNOWLEDGMENT

We are thankful to our Department of Computer Science and Engineering and NLP Club of Shahjalal University of Science and Technology, Sylhet 3114, Bangladesh, for the help they have always provided us with and for motivating this research. We would like to be very grateful to our honorable supervisor for his support and motivation.

REFERENCES

[1] B. Liu, "Opinion mining, sentiment analysis, opinion extraction," available at: https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html. (Accessed on 16 May 2019).

[2] M. A. Rahman and E. Kumar Dey, "Datasets for aspect-based sentiment analysis in bangla and its baseline evaluation," *Data*, vol. 3, no. 2, 2018. [Online]. Available: https://www.mdpi.com/2306-5729/3/2/15

[3] M. Nabil, M. Aly, and A. Atiya, "Astd: Arabic sentiment tweets dataset," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 2515–2519.

[4] M. S. Akhtar, A. Ekbal, and P. Bhattacharyya, "Aspect based sentiment analysis in Hindi: Resource creation and evaluation," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 2703–2709. [Online]. Available: https://www.aclweb.org/anthology/L16-1429

[5] A. K. Paul and P. C. Shill, "Sentiment mining from bangla data using mutual information," in *2016 2nd International Conference on Electrical, Computer Telecommunication Engineering (ICECTE)*, Dec 2016, pp. 1–4.

[6] S. Chowdhury and W. Chowdhury, "Performing sentiment analysis in bangla microblog posts," in *2014 International Conference on Informatics, Electronics Vision (ICIEV)*, May 2014, pp. 1–6.

[7] M. Al-Amin, M. S. Islam, and S. Das Uzzal, "A comprehensive study on sentiment of bengali text," in *2017 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, Feb 2017, pp. 267–272.

[8] R. Longadge and S. Dongre, "Class imbalance problem in data mining review," *arXiv preprint arXiv:1305.1707*, 2013.

[9] S. Symeonidis, "5 things you need to know about sentiment analysis and classification," available at: https://www.kdnuggets.com/2018/03/5-things-sentiment-analysis-classification.html. (Accessed on 16 June 2019).

[10] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

[11] F. Miedema, "Sentiment analysis with long short-term memory networks," *VRIJE UNIVERSITEIT AMSTERDAM*, vol. 1, 2018.

[12] S. Minaee, E. Azimi, and A. Abdolrashidi, "Deep-sentiment: Sentiment analysis using ensemble of cnn and bi-lstm models," *arXiv preprint arXiv:1904.04206*, 2019.

2019 International Conference on Electrical, Computer and
Communication Engineering (ECCE), 27-28 September 2019

Topic Modelling: A Comparison of The Performance of Latent Dirichlet Allocation and LDA2vec Model on Bangla Newspaper

Md. Hasan[1], Md. Motaher Hossain[2], Adnan Ahmed[3] and Mohammad Shahidur Rahman[4]
Department of Computer Science and Engineering
Shahjalal University of Science and Technology
Sylhet-3114, Bangladesh
Email: [1]mdhasansust@gmail.com, [2]md.motaherhossain04@gmail.com, [3]sust.adnan@gmail.com, [4]rahmanms.bd@gmail.com

***Abstract*—Topic modeling is a statistical data mining method for organizing the documents with variable contents under similar topics.It is utilized to reveal the concealed topics from an enormous collection of articles or documents.In this research, we have conducted an experiment on the effectiveness of two trending topic modeling methods on Bangla news corpus.Our target is to find an effective way of analyzing Bangla news documents and categorize them automatically for a recommendation system and searching. Different Models have been found effective for different languages because of their unique morphological structure.In this research LDA and its hybrid with word2vec known as lda2vec have been chosen as techniques to extract topics from Bangla news documents.LDA is a matrix factorization technique and as our observation in this research, this model is effective on Bangla despite the syntax difference with English.For the testing and implementation purpose, we developed a technique to find topics for not factorized documents from LDA and lda2vec.As our finding, lda2vec gave 85.66% accuracy over topics for test documents where accuracy for LDA was 62.45% Which makes lda2vec more reliable for the implementation purpose.**

***Index Terms*—Topic modeling, LDA, word2vec, lda2vec, Bangla Topic Modeling.**

I. Introduction

Over the last decade,the quantity and diversity of electronic documents have increased rapidly. With the growth of online newspapers, blogs and social sites, it became very hard to index and search from all the data available right now. It became a necessity to develop optimized techniques or tools to deal with automatically browsing, indexing,searching and organizing large collections of documents.One of the techniques that focus on the extraction of hidden meaning from documents, automatically analyzing and categorizing text data is topic modeling.

Topic modeling is a form of text mining, a way of identifying patterns in a corpus. It is a technique of generating a probability distribution of words over a set of documents.This process considers that documents are weighted combination of topics and topics are mixture of words according to their probability distribution. This distribution is made by analyzing a corpus and assigning weights for each word which is done by statistical topic modeling methods. They generate sets of topically-related words as 'topic' which can be associated with the documents of that corpus. This process provides the ratio of a topic in each document as documents could consist of multiple issues or genre. This also provides quantitative data which could be used to identify documents with a particular topic and generate visualization of the document set as a whole. Topic modeling technique used for developing an alternative way of searching, browsing and summarizing large document set. Statistical models are used for topic modeling to extract topics from a large document set and LDA [1] is one of the recent and effective models.

LDA is being used in different languages with necessary modification to satisfy morphological structure and research specific data analysis.

Mustakim Al Helal, Malek Mouhoub [2] have conducted research on topic modeling and document similarity measurement on Bangla articles using LDA and compared it similarity measurement accuracy with Doc2Vec.In [3],a topic modeling have been used on Bangla books and articles to extract features for supervised authorship attribution.

Tarek Kanan, Souleiman Ayoub [4] used RenA with ALDA, where ALDA is a modified version of LDA to support the Arabic morphological structure, stemming, stopword removal and normalization. to generate topics from Arabic news documents. They detected Arabic named entities to extract topics with Arabic Name Entity Recognizer (RenA),it has been described in Benajiba's paper [5].

K Shakeel, GR Tahir, I Tehseen [6] used GIBBS SAMPLING with LDA model. Gibbs sampling helped to generate a cycle of word distribution which is approximated from multivariate probability distribution [7] [8].From these word distribution LDA generates topic distribution.

Rifki Adhitama, Rahmat Gernowo, and Retno Kusumaningrum [9] have used Naive LDA with Ontology.They implemented LDA to form the cluster, Zhou similarity to measure between terms in a cluster and ontology to generate the label of the cluster.They measured the the weight of labels using Cohen's Kappa coefficients [10].

978-1-7281-5242-4/19

II. Background Study

This section provides a brief description of LDA, lda2vec and their process of extracting topics from a corpus.

A. *LDA*

LDA is a widely used topic modeling method in these days,it is a matrix factorization technique and a statistical model that extracts word sets as topics from a set of a text document.It takes input documents as fixed-length vectors (bag-of-words).As it is is a statistical model with the ability to a generate probability distribution of words, this model can be used to solve other ML problems beside topic modeling, like it could be used to extract feature from documents for authorship attribution [3].

If we Train LDA with a document set containing N articles and assume that those articles made of M topics,LDA will find M vectors of topic which could be distributed to explain the articles.If the data contains V words, then each topics will be a vector with V dimensions where the weight of each dimension points to the relevance of each word to the topic.An article could be made of 60 percent of "topic a" and 40 percent of "topic b".document vectors could contain some zeros,which means that all topics don't occur in each document.

The process of LDA could be visualized with a figure.In the "Fig. 1",the exterior box represents data, the box int the middle represents selected topics and their words for a document. M is document count and N is word count in the data.

Here,

α = probability of topic orientation for each document

β = probability of word collection for each topic

θ = the topic orientation for document d

z = the specific topic from topic orientation of document d

w = a word from the topic

B. *lda2vec*

lda2vec [11] is a combination of LDA and word2vec, it learns word vectors using skip-gram model along with topic vectors .Document vectors as a relation of words with documents and a word embedding as word to word relation have been combined to form lda2vec.

In LDA,a probability distribution of words is used to predict a topic, this could be used to predict a word after the occurrence of a related word. In word2vec, word vector from skip-gram is used to predict a context word. lda2vec leverages this word vector with a document vector and creates a context vector, which could be used to backtrack and predict word for each word in a document and generate word sets as topics.

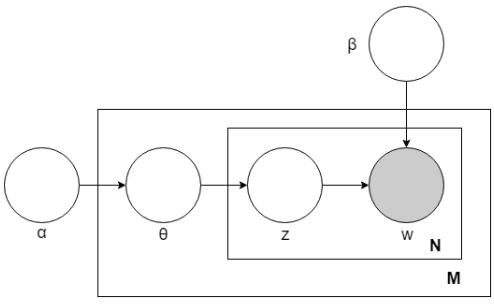


Fig. 1. Plate notation of the LDA model

In lda2vec, Skip-gram Negative-Sampling (SGNS) [12] has been altered to use document-wide feature vectors while concurrently studying the constant charging of document weights on topic vectors. The total loss term $\mathcal{L}$ in "(1)" is the sum of the Skipgram Negative Sampling Loss (SGNS) $\mathcal{L}_{ij}^{neg}$ with the addition of a Dirichlet-likelihood term over document weights,$\mathcal{L}^d$. The loss is conducted using a context vector,$\overrightarrow{c}_j$, pivot word vector $\overrightarrow{w}_j$, target word vector $\overrightarrow{w}_i$ and negatively-sampled word vector $\overrightarrow{w}_l$.

$$\mathcal{L} = \mathcal{L}^d + \sum_{ij} \mathcal{L}_{ij}^{neg} \tag{1}$$

$$\mathcal{L}_{ij}^{neg} = \log \sigma \left(\overrightarrow{c}_j . \overrightarrow{w}_i \right) + \sum_{l=0}^{n} \log \sigma \left(- \overrightarrow{c}_j . \overrightarrow{w}_l \right) \tag{2}$$

lda2vec makes a combination of the best parts of LDA and word2vec in a single model, this improves the quality of topic modeling.

III. Proposed Work

This Topic modeling system includes LDA and lda2vec models for extracting topics from news. The workflow contains different phases, including data collection, pre-processing, training, and predicting topics for test data. News articles have been parsed from online news portals and stored in text files for pre-processing.Stop words have been removed and bag-of-words has been created in the pre-processing stage. While training,Topics for all documents have been generated and weights for each word for each topic have been extracted. At the final step, topics for the new article have been predicted based on their word distribution.The workflow is represented in the "Fig. 2".

IV. Data Preparation

In this section, we present a brief description of the data-set we have used for the training purpose.

A. Data Sources

We have used a collection of Bangla news document. The documents were collected from various online Bangla news portals and online newspapers. This data set is made of 22,675 news articles, split into 9 different news types. It was created by PIPILIKA, this document set is being used for data mining experiments of Bangla text data.

This data could be categorized into 9 different news types.From contextual observation , Some types may describe related events (e.g. crime/accident), some could be totally different with different word sets (e.g. politics/sports).Table I, contains a brief overview of the data:

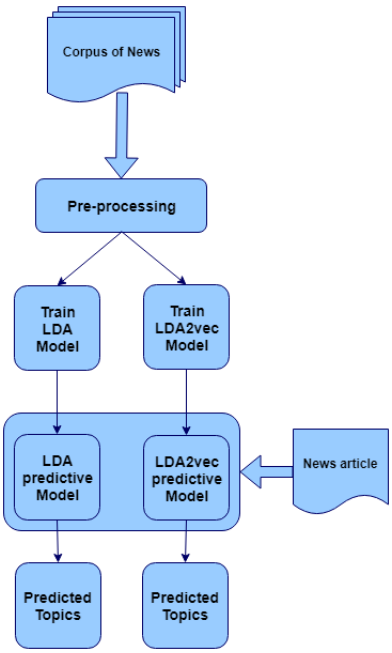


Fig. 2. System architecture

B. Tokenization

At first, all documents have been tokenized. It is a process where a stream of text is broken into words, phrases or other meaningful units. This group of units then becomes the element for further processing. This process has been done with simple python NLTK library. Punctuations have been removed from the sentences, then text stream has been split up by whitespace characters. white space characters are tabs, spaces, and carriage returns. Other non-word and non-number characters also have been removed from the tokenized data set.

C. Stop Words Removal

Stop words are words with few or no information which occurs quite frequently in documents. These are also called functional words. They connect the words inside a sentence and create a meaningful structure. They dont carry much meaning of their own. To clean the document set, a static list of 398 stop words has been collected, the list has been generated by Bangla language experts, it's available in this link link[1]

TABLE I
DATASET

News Categories	Document count
Accident	2500
Crime	2523
Entertainment	2814
International	2570
Politics	3284
Science	3336
Sports	2659
Weather	2919
Total(9 categories)	22,675

D. Finding meaningful words

There are a lot of words other than stop words which occur frequently in a text and doesnt contain meaningful relationship with the topic of the document. These words are not very useful for finding the inner meaning of the data. These words occur frequently in every document which means they are not attached to any particular topic. The words that occurred very little in documents also have to be removed.

The documents which became blank after removing all stop words and irrelevant words also have been removed from the process.

V. Experiment and Results Analysis

In this part, we have conducted an experiment on the selected models.First, both LDA and lda2vec models have been trained with our preprocessed data set.The models have generated sets of words as topics. Then the word sets have been tagged with topic names manually with the help of a volunteer.Then test data set have been prepared to analyze the performance of the models.

To identify the topics of the test documents using the models we have developed a technique of analyzing not factorized documents. LDA uses matrix factorization technique, it calculates weights for each word according to assigned topics on the documents. First, we have generated word weights for each word on a topic distribution by LDA and lda2vec and compared each word of the test documents against it, then analyzed the summation of the word weights for the documents to generate a topic weight for each document and chosen topics with positive weights for the document.

We have chosen 566 not factorized documents as test data.These documents have contained keywords which point to death or related events. Then we tested those documents with our models if those documents actually 'Crime' related documents or they belong to some other topics.

We have gotten multiple topics for most of the test documents.A volunteer has chosen topics from previously generated topic names for each test document manually,then checked how many topics actually belong to each document from the topics which was generated by the models for that document.From that observation, an accuracy chart has been created for each topic for both LDA and lda2vec model.

We have generated a chart which contains the ratio of how often a topic got tagged to documents and how often it actually belonged to those documents.

In the "Fig. 3" , we can see the accuracy for each topic generated by LDA is arbitrary, accuracy for some topics are very high, and for some topics frequently got tagged to the documents they dont belong. This makes the model quite unreliable.

[1]https://github.com/stopwords-iso/stopwords-bn/blob/master/stopwords-bn.txt

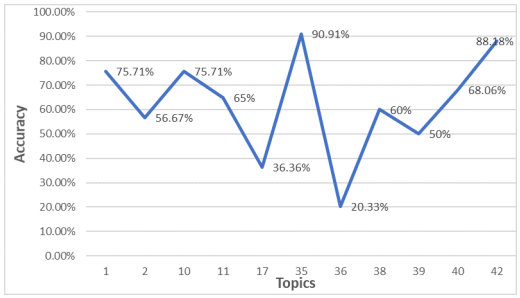


Fig. 3. Accuracy for topics in LDA

In the "Fig. 4", we can see the accuracy for each topic generated by lda2vec is well ordered, the accuracy of topics are from high to average. This makes the model reliable, useful and feasible for real-life projects.

Comparison between both models are shown in "Fig. 5".

From the average of accuracy generated for topics, we can show that

- Accuracy of LDA is 62.45%
- Accuracy of lda2vec is 85.66%.

As we discussed before, the result of LDA is not very reliable, The topics which have been generated by the model for documents not always accurate, some topics belong to the document and some are not. This is the main reason for decreasing the accuracy of this model.

On the other hand, lda2vec implements word2vec to generate a document vector which gives it more accuracy in the context of word-topic probabilistic distribution. This is the main reason for detecting topics which are most related to the test data. This is why its result is far more accurate than LDA.

To understand the improvement of these models, we can look at the output of some legacy topic identification processes that has been used for a long time.In [13], an experiment have been done to analyze 5 TID models with 1500 newspaper data containing 7 news categories.They have marked each news with multiple topic tags .The following table contains there accuracy of identifying all topics.

from the output, it is clear that the two modern topic modeling methods clearly outperforms these legacy systems.Although the data set is different, but we can get an understanding of the improvement in the process of identifying topics with LDA and lda2vec.

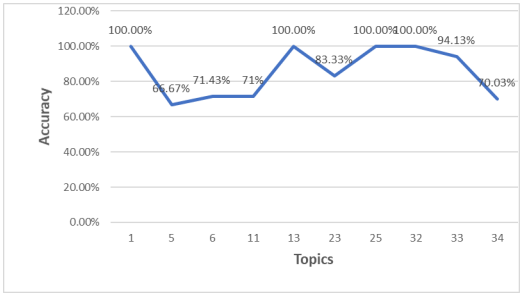


Fig. 4. Accuracy for topics in lda2vec

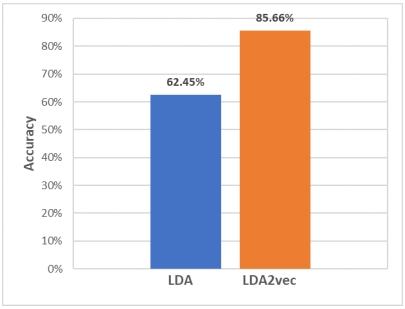


Fig. 5. Accuracy of models

TABLE II
TID PERFORMANCE ON NEWSPAPER

Models	Accuracy
Topic Unigram	45.7
Cache Model	48.8
TFIDF	29.9
Perplexity	47.7
Weighted Unigram	40.3

VI. Conclusion

This paper demonstrates an analysis of two latest topic modeling methods LDA and lda2vec and their performance on Bangla Language.In this research, a set of 22,675 Bangla news has been collected and preprocessed. The models have obtained topics for text documents and generated weight values for each term of the documents which have been used to predict the most relevant topics of test documents. LDAs accuracy was 62.45% and lda2vecs accuracy was 85.66%. From the result, it becomes clear that lda2vec outperforms LDA and it is more reliable for implementation purpose. This model could be used for the Bangla news recommendation system, text mining of massive Bangla news corpus and other statistical analysis of Bangla Document.

References

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[2] M. Mouhoub and M. Al Helal, "Topic modelling in bangla language: An lda approach to optimize topics and news classification." *Computer and Information Science*, vol. 11, no. 4, pp. 77–83, 2018.

[3] S. Phani, S. Lahiri, and A. Biswas, "A supervised authorship attribution framework for bengali language," *arXiv preprint arXiv:1607.05650*, 2016.

[4] T. Kanan, S. Ayoub, E. Saif, G. Kanaan, P. Chandrasekarar, and E. A. Fox, "Extracting named entities using named entity recognizer and generating topics using latent dirichlet allocation algorithm for arabic news articles," Department of Computer Science, Virginia Polytechnic Institute & State , Tech. Rep., 2015.

[5] Y. Benajiba, "Arabic named entity recognition." PhD dissertation, Universidad Politcnica de Valencia.Valencia, Spain, 2010.

[6] K. Shakeel, G. R. Tahir, I. Tehseen, and M. Ali, "A framework of urdu topic modeling using latent dirichlet allocation (lda)," in *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2018, pp. 117–123.

[7] W. R. Gilks, S. Richardson, and D. Spiegelhalter, *Markov chain Monte Carlo in practice*. Chapman and Hall/CRC, 1995.
[8] R. P. Dobrow, *Introduction to stochastic processes with R*. John Wiley & Sons, 2016.
[9] R. Adhitama, R. Kusumaningrum, and R. Gernowo, "Topic labeling towards news document collection based on latent dirichlet allocation and ontology," in *2017 1st International Conference on Informatics and Computational Sciences (ICICoS)*. IEEE, 2017, pp. 247–252.
[10] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and psychological measurement*, vol. 20, no. 1, pp. 37–46, 1960.
[11] C. E. Moody, "Mixing dirichlet topic models and word embeddings to make lda2vec," *arXiv preprint arXiv:1605.02019*, 2016.
[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.
[13] B. Bigi, A. Brun, J.-P. Haton, K. Smaïli, and I. Zitouni, "A comparative study of topic identification on newspaper and e-mail," in *Proceedings Eighth Symposium on String Processing and Information Retrieval*. IEEE, 2001, pp. 238–241.

International Conference on Bangla Speech and Language Processing (ICBSLP), 27-28 September 2019

Cricket Sentiment Analysis from Bangla Text Using Recurrent Neural Network with Long Short Term Memory Model

Md. Ferdous Wahid
Dept. of Electrical and Electronic Engineering
Hajee Mohammad Danesh Science and Technology University,5200
Dinajpur, Bangladesh
mfwahid26@gmail.com

Md. Jahid Hasan
Dept. of Electrical and Electronic Engineering
Hajee Mohammad Danesh Science and Technology University,5200
Dinajpur, Bangladesh
jahidnoyon36@gmail.com

Md. Shahin Alom
Dept. of Electrical and Electronic Engineering
Hajee Mohammad Danesh Science and Technology University,5200
Dinajpur, Bangladesh
ashahin200@gmail.com

***Abstract*— Nowadays, people used to express their feelings, thoughts, suggestions and opinions on different social platform and video sharing media. Many discussions are made on Twitter, Facebook and many respective forums on sports especially cricket and football. The opinion may express criticism in different manner, notation that may comprise different polarity like positive, negative or neutral and it is a challenging task even for human to understand the sentiment of each opinion as well as time consuming. This problem can be solved by analyzing sentiment in respective comments through natural language processing (NLP). Along with the success of many deep learning domains, Recurrent Neural Network (RNN) with Long-Short-Term-Memory (LSTM) is popularly used in NLP task like sentiment analysis. We have prepared a dataset about cricket comment in Bangla text of real people sentiments in three categories i.e. positive, negative and neutral and processed it by removing unnecessary words from the dataset. Then we have used word embedding method for vectorization of each word and for long term dependencies we used LSTM. The accuracy of this approach has given 95% that beyond the accuracy of previous all method.**

Keywords— sentiment analysis, natural language processing, deep learning, word embedding, RNN, LSTM.

I. INTRODUCTION

In present era, people across the globe express their opinions or feelings through social media and web on different entities such as events, products, social issues, organizations etc. Hence, in every instant massive amount of text data are generated on various entities over the Internet. By analyzing these data business organizations can understand the sentiment of people about their products and can find new opportunities, government can understand people perception about election and can manage their reputation, event organizer can understand people expectation on public events and so on. Thus, it is a high need to epitomize the unstructured data created by people over the social media and extract relevant insights in order to understand people thoughts. Therefore, Sentiment Analysis has become a major point of focus in the field of NLP which extract contextual mining from text data that conveys emotions, sentiments or opinions of an individual.

In recent times, Cricket has gained uttermost popularity in Bangladesh. So, people have diversified emotions for this sport. They like to express their opinions, emotions regarding to this sport most often through social network in Bangla language. By processing these reviews, it is possible to understand the sentiment of people for cricket. However, very few attempts were taken for sentiment analysis from Bangla text because of the unavailability of well-structured resources in Bangla language processing. Hence, Cricket sentiment analysis on Bangla text from real people sentiments for cricket has become an exciting field for us. Nowadays, Deep learning technique is widely used to analyze sentiment of text and has proven to be an effective tool in terms of accuracy as it considered past and future word with respect to target word for text classification. Thus, we are very much influenced to classify cricket sentiment from Bangla text using deep learning technique.

In this research, Recurrent Neural Network with LSTM model has been proposed to identify cricket sentiment from Bangla texts. We have collected real people sentiments about cricket from different social media and news portal and categorized into positive, negative or neutral. Then, vectorization of each word was performed by word embedding method and LSTM was used to achieve long term dependencies. Finally, the accuracy of 95% has attained in cricket sentiment analysis using the proposed model.

II. RELATED WORK

Sentiment analysis from Bangla text has become major point of focus in NLP for researchers with the increasing use of social media. Hasan et al. [1] proposed a model utilizing LSTM with binary cross-entropy and categorical cross-entropy loss function for sentiment analysis of Bangla and Romanized Bangla Text (BRBT). Sharfuddin et al. [2] developed an approach combining deep RNN with bidirectional LSTM to classify sentiment of Bengali text which achieved 85.67% accuracy on a dataset containing 10000 comments of Facebook status. Baktha et al. [3] have explored RNN architectures on three dataset and obtained best accuracy from Gated Recurrent Units (GRU) for sentiment analysis. Tripto et al. [4] suggested deep learning based models for detecting multilabel sentiment and emotions from Bengali YouTube comments. They used Convolutional Neural Network (CNN), LSTM, Support Vector Machine (SVM) and Naïve Bayes (NB) architectures to identify three (positive, negative, neutral) and five label (strongly positive, positive, neutral, negative, strongly negative) sentiment as well as emotions where they considered SVM and NB as their baseline methods. Term Frequency Inverse Document Frequency (TF-IDF) with n-gram tokens has been used to extract set of features from respective sentence. They got 65.97% and 54.24% accuracy for three and five labels

978-1-7281-5242-4/19

sentiments respectively. Sentiment polarity detection approach has been investigated on Bengali tweet dataset by Sarkar et al. [5] by applying multinomial NB and SVM. A character level supervised RNN approach was used to classify Bengali sentiment which is categorized as positive, negative and neutral [6]. An Aspect-Based Sentiment Analysis has been evaluated on Cricket comment in Bangla text in [7] where best accuracy has been found 71% using SVM classifier in their ABSA dataset. Shamsul et al. [8] employed SVM, Decision Tree and Multinomial NB for sentiment analysis on Bangladesh cricket comments. They got best accuracy using SVM which is 74%. So, we were influenced to develop a model for analyzing sentiment of real people by utilizing RNN with LSTM model.

III. Dataset Preparation

The preparation phases of the dataset were divided into two parts, i.e. A. Gathering of Bangla comments and B. Pre-processing of Bangla comments.

A. Gathering of Bangla comments

We have collected a dataset named 'ABSA' [7] that contains cricket related comments for cricket sentiment analysis from Bangla texts. The dataset comprised of 2979 data with 5 columns where we have selected only two columns, i.e. the comment column and the target column. The target column contains 3 classes including positive, negative and neutral. But, proposed LSTM-RNN model may create high variance problem on small dataset. To overcome this issue, we supplemented more data with existing ABSA datasets. The extended data were picked from various online resources like Facebook, YouTube, Prothom-Alo, BBC Bangla, Bdnews24.com and labelled them manually. Then, the extended ABSA dataset contains total 10000 comments where 8000 comments is separated for training and the rest of the comments is used for testing purpose.

B. Preprocessing of Bangla comments

Data pre-processing plays significant role in natural language processing (NLP) as the real-world data are messy and often contains error, unnecessary information and duplication. So, in order to generate good analytics results, all punctuation, unimportant words are removed, stemmed to their roots, all missing values are replaced with some values, case of text are replaced into a single one and mostly depending upon the requirement of the application. Therefore, we process our data step by step as it doesn't carry much weightage in context of the text. All preprocessed step is illustrated below.

1) Stopwords Removal: Stopwords refers to the most common words in a language. But these words have no impact on analysis sentiment of a sentence. The most common words such as এবং, এবার, এ, এটা, কী, হয়, র, পর, ওরা, কে, কেউ, ইত্যাদি have no impact on sentiment analysis using proposed model. But there some words such as না, নাই, নেই, নয় have important impact on negative sentiment and some words such as হ্যা, স্পষ্ট, করে, কাজ, কাজে have also important impact on positive sentiment. So, we list these positive and negative sentiment impact words from stopwords list as a whitelist. Then programmatically we have removed all stopwords from our dataset. We use two resources as a benchmark for removing Bangla stopwords.

2) Text Process: Text processing is of great importance in NLP task. The unwanted text such as Links, URLs, user tags and mention from comments, hash-tags, punctuation marks have no impact on sentiment analysis. Therefore, we remove these to give annotators an unbiased text. Only contents have given higher priority to make a decision based on three criteria positive, negative and neutral.

3) Name Process: Name process is another text data compression technique where mainly all proper and common noun of Bangla is substitute by common words. Example-

১. তামিম আজকে খুব সুন্দর ব্যাটিং করেছে।

২. লিটন দাশকে আজকে দলে নেয়া উচিত ছিল।

Here তামিম and লিটন are two different words but same context and also contribute similar impact on sentiment analysis using proposed LSTM-RNN model. So, we replace all proper noun with common word which does not affect the accuracy of the model but compress the dataset and makes dataset more robust and better for classification accuracy. Thus, we replace all country name such as (বাংলাদেশ, ভারত, পাকিস্তান, অস্ট্রেলিয়া) by a word "দেশ" and all players name including different spelling and nick name such as (তামিম, তামিম ইকবাল, মাশরাফি, ম্যাশ, সাকিব, বিরাট কোহলি, মুশফিকুর রহিম, মুশফিক) by word "সে" .

4) Manual validation: In manual validation, each extended data sample was manually annotated by three annotators into one of three categories: (i) positive (ii) negative and (iii) neutral. Each annotator validated the data without knowing decisions made by other. This ensures that the validations were unbiased and personal. Furthermore, elongated words often contain more sentiment information for multiclass categorizations. For example, "বাহ্হ্হহ অনেক ভালো!!" certainly provides more positive feelings. Therefore, instead of applying lemmatization we had kept elongated words.

IV. Methodology

Recurrent Neural Network (RNN) performed well enough on sequential input data like speech, music, text, name entity recognition etc. than convolutional neural network (CNN) by taking into consideration the current input as well as previous input. However, in long term dependencies it not generally works well due to vanishing and exploiting gradient problem [9]. Hence in order to learn long term dependencies, LSTM is simply added to the process input which enables RNN to remember their inputs over a long period of time. Here, this LSTM layer is fed with proper numerical vector representation of each word which is generated based on the word embedding. To generate word embedding, we have employed word2vec algorithm that formulates matrix of weight from text corpus. Finally, the output of LSTM layer is sent to fully-connected softmax layer to analysis sentiment of a comment.

A. Word Embedding

The preprocessed text data is tokenized first in order to split a sequence of sentence into smaller parts such as words. Then we implemented skip gram [10] model of word2vec algorithm for representing proper numerical vector of each

splitting word that can evaluate the similarity between words and also placed close together in the vector space which is computationally effective for learning word embedding than one-hot vector representation. The output of the word2vec model is called an embedding matrix. The complete word embedding process is shown in Fig. 1.

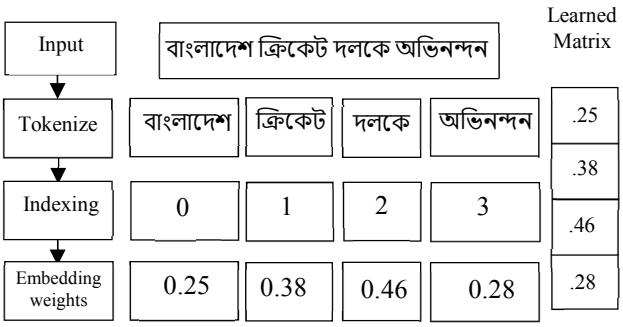


Fig. 1. Complete word embedding process

B. Model Architecture for sentiment analysis

The proposed architecture built to classify the sentiment of the cricket comments, consists of 4 layers. These layers are

1. Input layer (length 1)
2. Embedding layer (Output of length)
3. LSTM layer (100 neurons)
4. Output layer (103 neurons) (including 100 dense layers and 3 sigmoid layers)

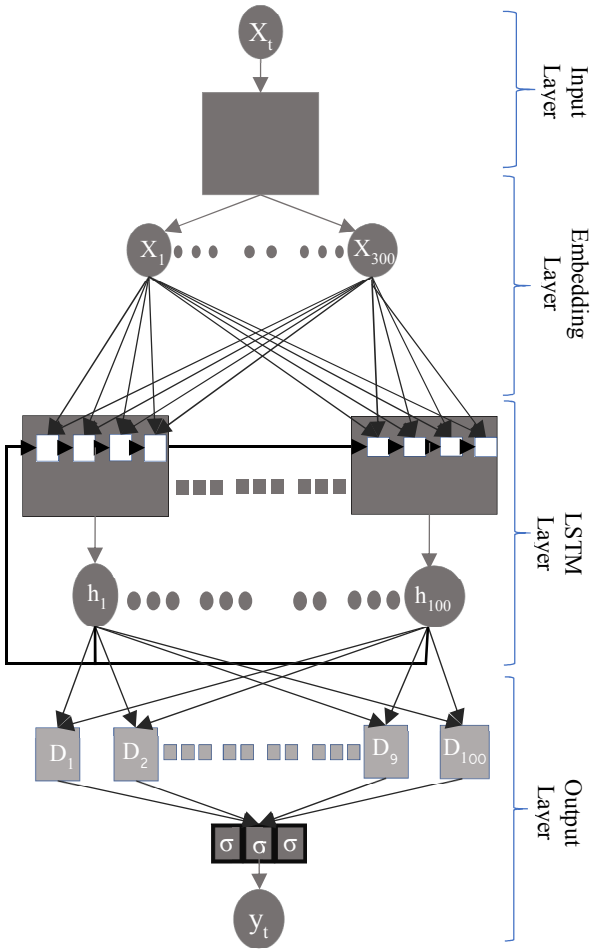


Fig. 2. LSTM network for cricket comments sentiment classification

A length of 42 words is used as a maximum word length of a comment and zero padding are added to the right of the comment when it is shorter than 42 words. The model takes input of 42 integers or vector of words, where each integer represents a word. So, there are 42-time steps, at each time step one word is given to the model. Then, the word is entered into the embedding layer with one neuron. The embedding layer transform the word into a numerical vector representation of length 300 (embedding size). The embedding weights are initially set to very small value and will update these weights using back-propagation during training. This way 300 featured value are created. Then, the output of embedding layer is fed into an LSTM layer with 100 neurons and each of the features value is multiplied by a weight of each LSTM cell, where LSTM cell contains four gates for long term dependencies memorization. Next those 300 weighted features and the output of the previous time step (output values from 100 neurons) is also used as an input for the LSTM cells. Finally, the weighted sum of dense layer outputs is taken as an input of softmax activation function where we predict the probability of cricket comments as positive, negative and neutral. The complete architecture is shown in Fig. 2.

V. Result and Discussion

We have used LSTM layer to construct and train many-to-one RNN architecture. The architecture takes sequence of words (sentence) as input and provides sentiment value (positive, negative or neutral) as output. We used data from the prepared dataset that contains 10,000 sentences in the ratio of 80:20 for training and test phase of proposed RNN-LSTM architecture respectively. During training phase, learning rate of the proposed architecture was set to a small value of 0.001 and different combinations of epochs and batch-size were used for attaining high prediction performance with minimum training time. The architecture attained best training accuracy after 15 epochs with batch size 30. Finally, the performance of the proposed architecture was evaluated on the test dataset where it obtained 95% prediction accuracy. The proposed model accuracy and loss graph was recorded while we train the model. Fig. 3 and Fig. 4 shows the model accuracy and loss graph respectively.

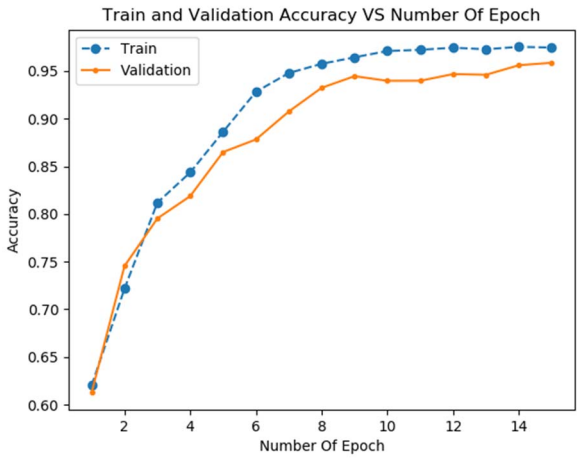


Fig. 3. Optimal accuracy of proposed model

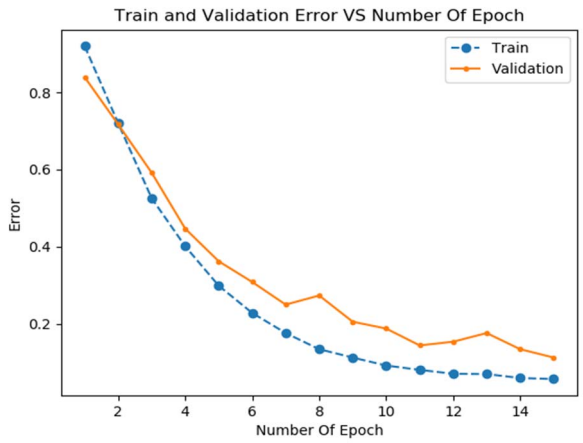


Fig. 4. Optimal loss of proposed model

Fig. 3 and Fig. 4 shows that the train and validation accuracy is being increased as well as is train and validation loss is being reduced respectively with respect to increasing the number of epochs. Example of some classification result is shown in TABLE I. The misclassification result is highlighted in the TABLE I using bold italic font.

TABLE I. SOME TEST RESULT

Sentence	Predict	Actual
বাংলাদেশ জিতবে ইনশাআল্লাহ।	positive	positive
টেষ্ট ক্রিকেটে রান আউট খুবই দুঃখজনক।	negative	negative
শাহারিয়ার নাফিস কে ও ফেরানো হোক।	positive	positive
ওরা ২০০ করছে তোমরা ১০০ করতে পারবে না	negative	negative
টস হারছে মানে, হারার সম্ভবনাই বেশী।	***neutral***	***negative***
মোসাদ্দেক, সাব্বির থেকে ভাল।	positive	positive
সেমিতে আমাদের প্রতিপক্ষ ভারত (প্রায় নিশ্চিত)।	neutral	neutral

Now a comparative study of classification accuracy among several models including proposed LSTM-RNN model is presented in Table II. It shows that the proposed architecture performs better than any other model of Bangla sentiment analysis, as RNN-LSTM model has a great competency to capture contextual information in more fine-grained way.

TABLE II. ACCURACY MATRIX AMONG DIFFERRENT BANGLA NLP TASK BASED ON MODEL PERFORMANCE

Ref No.	Dataset	Model	Prediction Accuracy
[7]	ABSA	SVM	71%
[8]	ABSA_EXTENDED	SVM	73.49%
Proposed method	ABSA_EXTENDED	LSTM	95%

VI. CONCLUSIONS

In this paper we present an approach to analyze sentiment of cricket comments in Bangla text. This model consists of a deep learning variant named RNN. For remembering the recurrent property and contextual meaning of a sentence we have used LSTM that makes the model very fruitful and produces a prediction result about 95%. Spell-checking and stemming is not included in preprocessing section of our collected dataset. In future, we will include more preprocessing steps along with these two in order to improve the structure of our dataset. We also plan to increase target class to make an accurate NLP model within this problem domain.

REFERENCES

[1] A. Hassan, M. Amin, A. Azad and N. Mohammed, "Sentiment analysis on bangla and romanized bangla text using deep recurrent models", 2016 International Workshop on Computational Intelligence (IWCI), 2016.

[2] A. Aziz Sharfuddin, M. Nafis Tihami and M. Saiful Islam, "A Deep Recurrent Neural Network with BiLSTM model for Sentiment Classification", 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), 2018.

[3] K. Baktha and B. K. Tripathy, "Investigation of recurrent neural networks in the field of sentiment analysis", 2017 International Conference on Communication and Signal Processing (ICCSP), 2017.

[4] N. I. Tripto and M. E. Ali, "Detecting Multilabel Sentiment and Emotions from Bangla YouTube Comments", 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), 2018.

[5] K. Sarkar and M. Bhowmick, "Sentiment polarity detection in bengali tweets using multinomial Naïve Bayes and support vector machines", 2017 IEEE Calcutta Conference (CALCON), 2017.

[6] M. S. Haydar, M. A. Helal, and S. A. Hossain, "Sentiment Extraction From Bangla Text : A Character Level Supervised Recurrent Neural Network Approach", 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), 2018.

[7] M. Rahman and K. E.Dey, " Datasets for Aspect-Based Sentiment Analysis in Bangla and Its Baseline Evaluation". Data, vol 3, issue 2, 2018.

[8] S. Arafin Mahtab, N. Islam and M. Mahfuzur Rahaman, "Sentiment Analysis on Bangladesh Cricket with Support Vector Machine", 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), Sylhet, pp. 1-4, 2018.

[9] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[10] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, "Distributed Representations of Words and Phrases and their Compositionality", NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems, vol 2, pp. 3111-3119, 2013.

Neural *vs* Statistical Machine Translation: Revisiting the Bangla-English Language Pair

Md. Arid Hasan
Cognitive Insight Limited, Bangladesh
arid.hasan.h@gmail.com

Firoj Alam
QCRI, Qatar
fialam@hbku.edu.qa

Shammur Absar Chowdhury
QCRI, Qatar
shchowdhury@hbku.edu.qa

Naira Khan
Dhaka University, Bangladesh
nairakhan@du.ac.bd

***Abstract*—Machine translation systems facilitate our communication and access to information, taking down language barriers. It is a well-researched area of Natural Language Processing (NLP), especially for resource-rich languages (e.g., language pairs in Europarl Parallel corpus). Besides these languages, there is also work on other language pairs including the Bangla-English language pair. In the current study, we aim to revisit both Statistical Machine Translation (SMT) and Neural Machine Translation (NMT) approaches using well-known, publicly available corpora for the Bangla-English (Bangla to English) language pair. We reported how the performance of the models differ based on the data and modeling techniques; consequently, we also compared the results obtained with Google's machine translation system. Our findings, across different corpora, indicates that NMT based approaches outperform SMT systems. Our results also outperform existing baselines by a large margin.**

***Index Terms*—Machine Translation, Bangla-to-English, Statistical Machine Translation, Neural Machine Translation, Bidirectional LSTM**

I. Introduction

With the emergence of neural sequence-to-sequence (seq2seq) [1] models, the task of automated translation from one language to another has undergone rapid advancement and is currently one of the most substantial fields in the NLP community. Despite the recent advancements in the field, the early concept of automated translation systems can be traced back to the 1950s [2] – motivated by translating information from foreign sources during the Cold War. Since then machine translation research has entered a different era, involving rule-based and SMT systems.

The idea of SMT, derived from information theory [3], is heavily dependent on the parameters of statistical models obtained from bilingual text corpora. These traditional data-driven SMT approaches, word- [4] and phrase-based [5] models, dominated the field of machine translation with their performance and made it useful for various applications.

With the recent advancement in neural networks and increased availability of computational resources, Neural Machine Translation (NMT) has gained significant momentum in the 20^{th} century [6]. In the last five years, it has reached state-of-the-art performance levels [7] and large scale deployment has also started. Particularly, the literature with NMT techniques report higher performance for resource-rich languages such as English to German [8] and English to French [9]. Unlike traditional techniques, the advantage of NMT is that it learns the mapping from input to output in an end-to-end fashion. In which it jointly learns the parameters in order to maximize the performance of the translation [10], [11].

While much of the literature reports the success of NMT approaches, some studies have also reported its limitations. The study of Koehn et al. [12] reports the limitations of current NMT systems including out-of-domain data, highly inflected categories lead to low-frequency words, amount of training data, long sentences and issues with word-alignment [12].

Based on the current developments and literature, in this study, we aim to revisit the different MT techniques for the *Bangla to English* (Bangla $\rightarrow$ English) *language pair*. Our contributions include, (1) conducting experiments using SMT and NMT approaches, (2) consolidating data from different sources and evaluating them to understand the effectiveness of these approaches, and (3) present a comparative analysis and provide future directions.

The motivation behind this study is two-faceted: (a) to enrich the Bangla NLP research community - as to the best of our knowledge, this is one of the first study to compare both SMT and NMT approaches for the Bangla to English language pair using publicly[1] available data; and (b) to understand what the necessary steps are and the resources required to outperform current state-of-the-art systems for the Bangla to English language pair.

For the current study, we use a bidirectional LSTM (BiLSTM) network to determine the state-of-the-art performance in comparison with other language pairs. On the other hand, our interest in using SMT for comparison, is due to its success in various MT systems [13], for different language pairs such as English to German, English to French, English to Spanish, etc. As for evaluating the quality of our designed MT models, we reported a BLEU (bilingual evaluation understudy) score.

The structure of this paper is as follows. Section II, provides a brief overview of the existing work on Bangla MT systems. In Section III, we discuss the datasets that we use in this study. We present the approaches that we use for our experiments in Section IV. Following that in Section V, we discuss the details of the experiments and report the results. In Section VI, we

978-1-7281-5242-4/19 ©2019 IEEE

[1]Except for Penn Treebank Bangla-English parallel corpus, which can be accessed by requesting to banglanlp@gmail.com.

discuss the results of the experiments. Finally, we conclude our work in Section VII.

II. Related Work

There have been many endeavors for both Bangla to English and English to Bangla machine translation systems, which include the development of a Bangla-English parallel corpus, evaluation, designing rule-based and automated systems by examining various machine learning algorithms [14]–[18].

In [19], the authors used the Indic Languages Multilingual Parallel Corpus [20] and achieved a BLEU score of 13.98 using Many-to-One phrase-based SMT. The experimental settings consist of a maximum sentence length of 50, a 3-gram language model, and the grow-diag-final and heuristic for extracting phrases [21]. In another study, authors designed a set of rules by analyzing Bangla and English grammars [16]. The rule-based system translated 25 out of 27 sentences correctly and the other two sentences are 75% and 33% correct respectively. In [15], authors report a phrase-based system, which used the EMILLE[2] and KDE4[3] corpora and achieved an overall BLEU score of 11.7 with the use of 8-gram language model, Moses, GIZA++, MERT, and preposition handling.

In [17], the authors used Context-Free-Grammars (CFGs) for English to Bangla machine translation, which uses morphological analysis, syntactic transfer, and sentence construction rules. In another study [14], the authors report active learning strategies for SMT and achieved a maximum BLEU score of $0.057(\sim 5.7\%)$ for Bangla to English machine translation using Hansardsand EMILLE corpora. In [22], authors report LSTM-based seq2seq model with attention mechanism and achieved a BLEU score of 10.92 with an accuracy of 20.5%. The most recent and notable work for the Bangla-English language pair has been done by Mumin et al. [23], where authors proposed a phrase-based SMT based solution, which is experimented and evaluated using one of the largest Bangla-English parallel corpora [24]. The study used log-linear phrase-based SMT, which outperform other phrase-based SMT techniques. The authors also highlight the complexities and the challenging issues that need to be addressed in future research.

Another notable work, for Bangla-English language pair, can be found in [25]. The authors, in the study, investigated both SMT and NMT results on Bangla English language pair. The study reports BLEU score of 16.56 using vanilla phrasal approach with word break, and 20.23 by using NMT approach with synthetic data augmentation for Bangla to English language pairs. Unfortunately, these results, in [25], are not directly comparable to ours, due to difference in dataset and lack of access to it.

Current state-of-the-art performance for resource-rich language pairs are relatively higher compared to the Bangla-English language pair. In [18], authors report an NMT approach, which jointly learns to align and translate using a RNN Encoder-Decoder. The reported BLEU score is 28.45 on English-French parallel corpora with the of maximum sentence length 50.

Our study differs from the reported studies in a way that we compare both SMT and NMT based approaches and present the utility of a sequence to sequence BiLSTM approach. From the evaluation across different corpora we report that our system outperforms existing baselines [19], [23] on the Bangla to English language pair.

III. Data

For the experiment we used corpora that are obtained from different sources, discussed below.

- Indic Languages Multilingual Parallel Corpus (ILMPC): This is publicly available through WAT [20], which has been made available in the Workshop on Asian Translation (WAT). The Indic Languages Multilingual Parallel Corpus consists of 7 parallel languages. The text in the Indic Languages Multilingual Parallel Corpus is collected from OPUS[4]. The ILMPC Bengali-English parallel corpus consists of $\sim$337K parallel sentences in the training set, 500 sentences in the development set and $1,000$ sentences in the test set. More details about these corpora can be found in [20].
- Six Indian Parallel Corpora [26] (SIPC): Six Indian Parallel Corpora consist of six languages. These parallel sentences were collected from the top-100 most-viewed documents from each language's Wikipedia [26]. The training, development and test sets consists of $\sim$20K, 914 and $1K$ parallel sentences, respectively.
- Penn Treebank Bangla-English parallel corpus (PTB): This corpus has been developed by the Bangladesh team of the PAN Localization Project [5], in which source English sentences have been collected from the Penn Treebank corpus. The dataset has been translated by expert translators, which consists of $1,313$ English-Bangla sentence pairs.
- SUPara Corpus [27], [28]: This corpus has been developed by the Shahjalal University of Science and Technology (SUST), in which sentences consist of different genres such as Literature, Journalistic, Instructive, Administrative, and External Communication. The publicly available version (SUPara0.8M) of this corpus contains $\sim$70.5K Bangla-English sentence pairs [27].

IV. Methodology

For our comparative study, for Bangla to English (BN $\rightarrow$ EN) language pair, we chose to use two different approaches of MT: *SMT* and *NMT*. As for SMT experiments, we used Moses statistical MT tool [21] and the NMT experiments are conducted using the OpenNMT toolkit [29]. We ran the initial experiments using training and development data sets and then, we evaluated the system using the test set. The performance was computed using BLEU and NIST scores [30]. Due to limited space, we are not reporting the NIST score in this paper.

[2]Corpus developed by Lancaster University, UK, and the Central Institute of Indian Languages (CIIL), Mysore, India.

[3]The corpus has been developed under OPUS http://opus.nlpl.eu

[4]http://opus.nlpl.eu

[5]http://www.panl10n.net

A. Preprocessing

For the SMT experiment, we tokenized the corpus using the Moses tokenizer. As a part of the tokenization process, all English sentences were transformed into lowercase. We limited our sentence length to 40, and removed longer sentences to reduce computational time for word-alignment process [13].

For NMT experiments, we tokenized the Bangla sentences using our developed Bangla tokenizer and the English sentences were transformed into lowercase. We did not tokenize English sentences. Then, we preprocessed our corpus using the OpenNMT preprocessing script. In OpenNMT preprocessing step, we set our sentence length limit to 50 and removed longer sentences to get better translation quality [12]. We split the sentences into tokens and tokens were then transformed into tensor values. Using these tensor values, we generated training, validation, and vocabulary files to train our NMT models.

B. Training

1) SMT: For the SMT training, we used the SRILM tool for building our Language Model (LM) and the Moses decoder for training. In language modeling, we used google one billion word [31], English monolingual data from ILMPC Corpus, English monolingual sentences from Europarl and News Commentary[6] corpora. Our language model contains approximately 1.5B word tokens. We trained 3-gram and 5-gram language models using the SRILM toolkit [32]. We used GIZA++ for word alignment in order to train a phrase-based SMT model.

2) NMT: For the NMT training, we used the BiLSTMs network-based approach. LSTMs [33] is a form of Recurrent Neural Network (RNN), which is widely used for capturing long-term dependencies. To predict the output of an input sequence RNNs capture all previous information in a memory cell, which is limited in predicting the output from a very long distance. To overcome this limitation, LSTMs were introduced, which consists of input, output and forget gates and are capable of capturing the long-term dependencies. We initialized hidden layer 500, word embedding size for both source and target 500, number of layers 2, and saved our models in every 10,000 steps. During the training, we initialized the model parameters using pre-trained word-embeddings. For English, we used the Glove model [34] and for Bangla we used the word2vec model [35]. The pre-trained embedding dimension is 300 for both the models. To avoid overfitting of the model we used a dropout rate of 0.3.

V. Experiments and Results

To evaluate the performances of the (BN $\rightarrow$ EN) models, we used Moses scripts to calculate the BLEU score [30].
We conducted two different sets of experiments:

1) *Exp Setting 1:* In this experiment we use ILMPC, SIPC, and PTB corpora. We mainly use ILMPC corpus for training, and evaluation. In order to improve the performance we merged all the training sets from these datasets into a training set. We used the ILMPC development set and test set in order to optimize the parameters and evaluate the systems, respectively. For the test setting, we removed all the sentences, from the Bangla parallel set, that contained any English words, resulting in 956 sentences instead of 1000 in the ILMPC test set. We have used this dataset, containing a total of $346,845$ parallel sentences (in Table I), to run the experiments using both SMT and NMT based approaches.
2) *Exp Setting 2:* For this experiment we used the SUPara corpus. We maintained the official training (SUPara0.8M), development (SUParadev2018) and test (SUParatest2018) set, which are publicly available [27], [28], as presented in Table I. From our experimental study (Exp Setting 1), we realized that NMT performs better, therefore, we only conducted experiments using NMT with this dataset.

Table I: Training, development and test data statistics. Bangla (BN), English (EN), Merged - merged dataset for Bangla to English (BN $\rightarrow$ EN) translation.

Data Set	# of Sent	# of Tokens	Source
Train	346,845	2.92M (BN), 2.42(EN)	Merged
Development	500	8,838 (BN), 7,762 (EN)	dev set (ILMPC)
Test	956	15,528 (BN), 13,612 (EN)	test set (ILMPC)
Train	70,861	886,337 (BN), 995,871 (EN)	SUPara0.8M
Development	500	9,157 (BN), 10,817 (EN)	SUParadev2018
Test	500	9,141 (BN), 10,819 (EN)	SUParatest2018

A. Results on Exp Setting 1:

In Table II, we present the performances of SMT and NMT systems including the baseline and Google Translator's results on the test set of our first experimental setting. From the results, we observe that with our combined dataset along with 3- and 5-gram language models our system outperforms baseline results. Comparing the results with the single dataset (row 2-3), with our merged data, we observe that there is a reasonable improvement in performance. From the results, it is also evident that a 5-gram language model is better than a 3-gram one. Note that in our merged dataset experiment, we retrain LM by combining the three datasets mentioned in Section IV-B1. Then, we used the Moses tokenizer to preprocess them. We realized that the tokenizer plays a significant role in performance, therefore, we aim to improve it in our future study.

The NMT results for single and merged datasets are significantly higher. We only report results with the combined dataset due to limited space, which proves the effectiveness of NMT results and it also corroborates current state-of-the-art.

For comparison, we also report the results of Google Translator, which is lower than all the reported results in Table II.

B. Results on Exp Setting 2:

In Table III, we report results on the SUPara test set (SUParatest2018). As mentioned earlier, we only used the SUPara training set (SUPara0.8M) to train the model. From

[6]http://www.statmt.org/wmt09/translation-task.html

Table II: Results on the test set for the Bangla to English (BN → EN) language pair. Indic Languages Multilingual Parallel Corpus (ILMPC), Penn Treebank Bangla-English parallel corpus (PTB), Six Indian Parallel Corpora (SIPC). EN-Emb: Glove Pretrained Embeddings, BN-Emb: Bangla Pretrained Embeddings.

Training Set	Experiments	BLEU
	google-translator	12.59
	SMT	
ILMPC	Baseline [19]	*14.17*
ILMPC	3-gram LM	13.24
ILMPC	5-gram LM	13.49
ILMPC + SIPC + PTB	3-gram LM	**14.61**
ILMPC + SIPC + PTB	5-gram LM	**14.82**
	NMT	
ILMPC + SIPC + PTB	BiLSTM	**15.24**
ILMPC + SIPC + PTB	BiLSTM + BN-Emb	**15.56**
ILMPC + SIPC + PTB	BiLSTM + BN-Emb, EN-Emb	**15.62**

the results it is evident that we obtain a higher performance gain using different NMT based approaches. Interestingly, Google Translator's results are far better than any other results including ours. We observe that the SUPara dataset (SUPara0.8M) are cleaner and simpler than the ILMPC dataset on which Google Translator performs better.

Table III: NMT results on the SUPara corpus for the Bangla to English (BN → EN) language pair. Trained using the SUPara training set (SUPara0.8M) and tested on the SUPara test set (SUParatest2018).

Training Set	Experiments	BLEU
	SMT	
SUPara0.8M	shu-torjoma [23]	*17.43*
	NMT	
SUPara0.8M	BiLSTM	**17.98**
SUPara0.8M	BiLSTM + BN-Emb	**19.28**
SUPara0.8M	BiLSTM + BN-Emb, EN-Emb	**19.76**
	Google Translator	***28.09***

VI. Discussion

To understand the performance of the models, we conducted a detailed error analysis comparing the source (S), automatic (A) and gold (G) translation of the target sentences. Our exploration indicates that one of the most common source of mismatch pairs are due to insertion or deletion. For example, **(S)**: মোরি, তাঁর সাথে আমি কথা বলব। is translated to **(A)**: '*morrie, i'll talk to him.*' instead of **(G)**: '**morrie, i'll go talk to him.**'. Here the word 'go' is deleted from the output. However, it is to be noted that even though the above sentence pairs are said to be mismatched, the semantic meaning remains intact and represent the source correctly.

Unlike the above benign error, there are translated instances that changes the meaning of the source sentences completely, like: (**S**: ভারতের কোন জায়গায় রকেট নিয়ন্ত্রণের ব্যবস্থা প্রস্তুত হয়? , **G**: **which is the place in india that manufactures rockets?**) → (**A**: *any place in india is prepared to control the rules of any kind?*). These types of errors are mostly observed in sentences due to the presence of rare words (e.g. রকেট - 'rocket'). While studying the causes, we realized that the datasets contain many rare (low frequency) words specially in the Bangla training set, in comparison to the English one and is one of the main reasons for lower performance of the model. We also observed that the quality of the translation is a big factor in these pairs (see the gold translation, G, in the above example) and an important factor for evaluating the performance.

Along with the aforementioned issues, other challenges for the Bangla-English pair are: i) morphological richness [36], results in highly-inflected words, ii) limited resources (e.g., amount of training data covering different domains) among others including the issues highlighted in [23].

Compared to the resource-rich languages, the difference in performance for the Bangla to English language pair is very high. As seen in [37], the performance of English-French MT using the WMT 2014 English-French dataset with a transformer model has a BLEU score of 41.0. This exhibits that more effort is required for Bangla-English machine translation research.

VII. Conclusions

In this study, we explored machine translation approaches for the Bangla to English (BN → EN) language pair. We compared the performance of SMT (phrase-based) vs NMT based approaches, in which we observed that NMT outperforms SMT based techniques. Our preliminary experiments outperform existing baselines on both the ILMPC and the SUPara test sets. Compared to the Google Translator, our system performs better on the ILMPC data set but not on the SUPara dataset. Our study reveals that tokenization is an important step and a well-designed tokenizer is necessary for Bangla. For NMT based techniques, its performance is highly dependent on parameter optimization and architecture, which we also plan to explore in the future.

Acknowledgment

The research leading to the results has been supported and funded by Cognitive Insight Limited – https://cogniinsight.com/.

References

[1] O. Kuchaiev, B. Ginsburg, I. Gitman, V. Lavrukhin, J. Li, H. Nguyen, C. Case, and P. Micikevicius, "Mixed-precision training for nlp and speech recognition with openseq2seq," 2018.

[2] T. Poibeau, *Machine translation*. MIT Press, 2017.

[3] P. Brown, J. Cocke, S. D. Pietra, V. D. Pietra, F. Jelinek, R. Mercer, and P. Roossin, "A statistical approach to language translation," in *Proc. of the 12th conference on Computational linguistics*. Association for Computational Linguistics, 1988, pp. 71–76.

[4] S. Vogel, H. Ney, and C. Tillmann, "Hmm-based word alignment in statistical translation," in *Proc. of the 16th conference on Computational linguistics*. Association for Computational Linguistics, 1996, pp. 836–841.

[5] P. Koehn, *Statistical machine translation*. Cambridge University Press, 2009.

[6] A. Waibel, A. N. Jain, A. E. McNair, H. Saito, A. Hauptmann, and J. Tebelskis, "Janus: A speech-to-speech translation system using connectionist and symbolic processing strategies," in *Proc. of the ICASSP*, 1991, pp. 793–796.

[7] S. Jean, O. Firat, K. Cho, R. Memisevic, and Y. Bengio, "Montreal neural machine translation systems for wmt-15," in *Proc. of the 10th WSMT*. Lisbon, Portugal: Association for Computational Linguistics, September 2015, pp. 134–140. [Online]. Available: http://aclweb.org/anthology/W15-3014

[8] S. Jean, K. Cho, R. Memisevic, and Y. Bengio, "On using very large target vocabulary for neural machine translation," *arXiv preprint arXiv:1412.2007*, 2014.

[9] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba, "Addressing the rare word problem in neural machine translation," *arXiv preprint arXiv:1410.8206*, 2014.

[10] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[11] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[12] P. Koehn and R. Knowles, "Six challenges for neural machine translation," *arXiv preprint arXiv:1706.03872*, 2017.

[13] P. Koehn and J. Schroeder, "Experiments in domain adaptation for statistical machine translation," in *Proc. of the 2nd WSMT*, 2007, pp. 224–227.

[14] G. Haffari, M. Roy, and A. Sarkar, "Active learning for statistical phrase-based machine translation," in *Proc. of the NACL*. ACL, 2009, pp. 415–423.

[15] M. Z. Islam, J. Tiedemann, and A. Eisele, "English to bangla phrase-based machine translation," in *Proc. of the 14th EACL*, 2010.

[16] J. Francisca, M. M. Mia, and S. M. Rahman, "Adapting rule based machine translation from english to bangla," *Indian Journal of Computer Science and Engineering (IJCSE)*, vol. 2, no. 3, pp. 334–342, 2011.

[17] S. S. Ashrafi, M. H. Kabir, M. M. Anwar, and A. Noman, "English to bangla machine translation system using context-free grammars," *International Journal of Computer Science Issues (IJCSI)*, vol. 10, no. 3, p. 144, 2013.

[18] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[19] T. Banerjee, A. Kunchukuttan, and P. Bhattacharyya, "Multilingual indian language translation system at wat 2018: Many-to-one phrase-based smt," in *Proc. of the 5th Workshop on Asian Translation*, 2018.

[20] T. Nakazawa, S. Kurohashi, S. Higashiyama, C. Ding, R. Dabre, H. Mino, I. Goto, W. P. Pa, A. Kunchukuttanand, and S. Kurohashi, "Overview of the 5th Workshop on Asian Translation," Tech. Rep., 2018. [Online]. Available: http://www2.nict.go.jp/astrec-att/

[21] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens *et al.*, "Moses: Open source toolkit for statistical machine translation," in *Proc. of the 45th ACL*, 2007, pp. 177–180.

[22] N. F. Liu, J. May, M. Pust, and K. Knight, "Augmenting statistical machine translation with subword translation of out-of-vocabulary words," *arXiv preprint arXiv:1808.05700*, 2018.

[23] M. A. A. Mumin, M. H. Seddiqui, M. Z. Iqbal, and M. J. Islam, "shutorjoma: An english↔bangla statistical machine translation system," *Journal of Computer Science (Science Publications)*, 2019.

[24] M. A. Al Mumin, A. A. M. Shoeb, M. R. Selim, and M. Z. Iqbal, "Supara: A balanced english-bengali parallel corpus," *SUST Journal of Science and Technology*, pp. 46–51, 2012.

[25] S. Dandapat and W. Lewis, "Training deployable general domain mt for a low resource language pair: English–bangla," 2018.

[26] M. Post, C. Callison-Burch, and M. Osborne, "Constructing parallel corpora for six indian languages via crowdsourcing," in *Proc. of the 7th WSMT*. ACL, 2012, pp. 401–409.

[27] M. A. Al Mumin, M. H. Seddiqui, M. Z. Iqbal, and M. J. Islam, "Supara0.8m: A balanced english-bangla parallel corpus," 2018. [Online]. Available: http://dx.doi.org/10.21227/gz0b-5p24

[28] M. A. Al Mumin, S. M. H., M. Z. Iqbal, and M. J. Islam, "Supara-benchmark: A benchmark dataset for english-bangla machine translation," 2018. [Online]. Available: http://dx.doi.org/10.21227/czes-gs42

[29] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush, "Opennmt: Open-source toolkit for neural machine translation," *arXiv preprint arXiv:1701.02810*, 2017.

[30] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: A method for automatic evaluation of machine translation," in *Proc. of the 40th ACL*, ser. ACL '02. Stroudsburg, PA, USA: ACL, 2002, pp. 311–318. [Online]. Available: https://doi.org/10.3115/1073083.1073135

[31] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, "One billion word benchmark for measuring progress in statistical language modeling," *arXiv preprint arXiv:1312.3005*, 2013.

[32] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *7th ICSLP*, 2002.

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[34] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. of EMNLP*, 2014, pp. 1532–1543.

[35] F. Alam, S. A. Chowdhury, and S. R. H. Noori, "Bidirectional lstms—crfs networks for bangla pos tagging," in *Proc. of ICCIT*. IEEE, 2016, pp. 377–382.

[36] M. A. Hasan, F. Alam, and S. R. H. Noori, "A collaborative platform to collect data for developing machine translation systems," in *Proc. of International Joint Conference on Computational Intelligence*. Springer, 2020, pp. 407–416.

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

International Conference on Bangla Speech and Language Processing(ICBSLP), 27-28 September, 2019

Intrinsic Evaluation of Bangla Word Embeddings

Nafiz Sadman*
Computer Science & Engineering
Independent University, Bangladesh
Dhaka, Bangladesh
nafizsadmanpreetom@gmail.com

Akib Sadmanee*
Computer Science & Engineering
Independent University, Bangladesh
Dhaka, Bangladesh
akibsadmanee@gmail.com

Md. Iftekhar Tanveer
Comcast Applied AI Lab
Washington DC 20005. USA
mdiftekhar_tanveer@comcast.com

Md. Ashraful Amin
Computer Science & Engineering
Independent University, Bangladesh
Dhaka, Bangladesh
aminmdashraful@iub.edu.bd

Amin Ahsan Ali
Computer Science & Engineering
Independent University, Bangladesh
Dhaka, Bangladesh
aminali@iub.edu.bd

Abstract—Word embeddings are vector representations of word that allow machines to learn semantic and syntactic meanings by performing computations on them. Two well-known embedding models are CBOW and Skipgram. Different methods proposed to evaluate the quality of embeddings are categorized into extrinsic and intrinsic evaluation methods. This paper focuses on intrinsic evaluation - the evaluation of the models on tasks, such as analogy prediction, semantic relatedness, synonym detection, antonym detection and concept categorization. We present intrinsic evaluations on Bangla word embedding created using CBOW and Skipgram models on a Bangla corpus that we built. These are trained on more than 700,000 articles consisting of more than 1.3 million unique words with different embedding dimension sizes, e.g., 300, 100, 64, and 32. We created the evaluation datasets for the above-mentioned tasks and performed a comprehensive evaluation. We observe, word vectors of dimension 300, produced using Skipgram models, achieves accuracy of 51.33% for analogy prediction, a correlation of 0.62 for semantic relatedness, and accuracy of 53.85% and 9.56% for synonym and antonym detection 9.56%. Finally, for concept categorization the accuracy is 91.02%. The corpus and evaluation datasets are made publicly available for further research.

Keywords— Word embedding, CBOW, Skipgram, Intrinsic evaluation, Bangla corpus

I. INTRODUCTION

Word embeddings are numerical representations of the words constructed from the "distributional hypothesis" [1], which defines that words that occur in the same context tend to exhibit similar semantic meaning. It can capture the semantic and syntactic structures of words and is found to improve performances in a number of Natural Language Processing (NLP) tasks [2]. These vectors have many features which can be used in applications ranging from information retrieval, document classification to question answering, named entity recognition, and parsing. There exists a number of word embedding techniques, e.g., word2vec [3], GloVe [4], C&W [5], H-PCA [6], TSCCA [7], and Sparse random projection [8]. In [3] Mikolov *et al*. introduced two of the most popular embedding methods – word2vec using Continuous Bag of Words (CBOW) and Skipgram models. CBOW model predicts a word from the context of that word. On the other hand, Skipgram model aims to predict neighboring words from a given word. Despite the vast use of these word embeddings, the main importance is defined by the "linguistic regularities and patterns" which the vectors encode [2]. These can be represented as linear translation of the embedding vectors.

One popular example is vec("রাজা") + vec("নারী") - vec("পুরুষ") is closer to vec("রাণী"). Thus the quality of a word embedding method can be defined on the contextual similarity of vector representations of two words.

There are two approaches to understand the quality of word embedding models, namely extrinsic and intrinsic evaluations. In extrinsic evaluations one measure the quality by measuring the performance matrices for specific downstream natural language processing tasks that uses the word embeddings. On the other hand, in intrinsic evaluations semantic and syntactic relationship between words are directly measured [9]. For example, in [9], the extrinsic evaluation is done by measuring the performance in two downstream tasks (namely, Noun phase chunking and Sentiment classification) using a pre-trained word embeddings. As for intrinsic evaluation, Baroni *et al.* [10] defined five benchmark tasks - semantic relatedness, synonym detection, selectional preferences, concept categorization and completing analogy. Yin *et al.* [5] has also demonstrated intrinsic evaluation on word relatedness. However, to our knowledge, no definite dataset or evaluation protocol exist for the intrinsic evaluation of Bangla word embeddings. There has been some recent work on word embedding applied to downstream tasks and only a few on evaluating word embeddings. For example, Ismail *et al.* [11] evaluated the performance of different word embedding models in terms of training time and cluster quality. However, no detailed description was provided about the procedure of the experiments and the evaluation dataset is not available. In [12], the authors evaluated sentence level embedding. They chose to average the word vectors for the words present in a sentence to generate the sentence level embedding and compare these with embeddings for another sentence. But this work does not evaluate word embeddings. Different Bangla word embedding models have been used to perform several downstream language processing tasks e.g. evaluating effects on Authorship Attribution of Bangla Literature [13], document classification [14]-[15], Sentiment Analysis [16]-[17], sentence classification [18], word level language identification in [19] and Named entity recognition [20]. All these works only addresses extrinsic evaluations of downstream tasks on Bangla word embeddings. However, performing intrinsic evaluation is important. Schnabel *et al.* point out in [9], although extrinsic evaluation can show the performance of word embedding for certain tasks, it cannot be

* Both authors contributed equally in this paper

978-1-7281-5242-4/19 ©2019 IEEE

used to evaluate the general quality of word embedding models.

In this paper, we therefore, present intrinsic evaluation for Bangla word embeddings. The methodology closely follows the evaluation methodology presented in [9]. However, there are a couple of challenges that need to be solved. First, there exists no evaluation datasets for Bangla. Also, there exists no pre-trained word embeddings for Bangla. Our goal is to close these gaps in Bangla NLP research. The contributions of our paper are as follows:

- We propose five intrinsic evaluation datasets and make them publicly available for future research. These consists of Bangla translations of familiar datasets such as wordsim353, portion of Analogy dataset of Mikolov *et al.*, as well as new datasets for Concept categorization, Synonym detection and Antonym detection tasks.
- We provide a Bengali corpus which is built with web-scraped data from different sources, mainly blogs or Bangla-Wikipedia. These are available at our github repository[1].
- We use the python framework gensim[2] to construct word embedding using CBOW and Skipgram model and provide a comprehensive evaluation of the embedding.

II. Constructing the Corpus

In this section we describe in detail the Bangla corpus that we constructed. Our corpus comprises of public dataset and web scraped data. We scraped blog posts, Bangla Wikipedia, e-books on Bangla novels, stories etc. and used a public dataset available in Kaggle [21] that contains articles of Bangla newspapers. Though a large portion of the data came from a single blog, it encapsulates writings on 24 different categories, i.e., story, poem, book review, higher education, etc.

Table I presents the details of the data that we used to create our corpus. As we can observe the majority of articles are blog posts.

TABLE I. Details of the datasets used for corpus building

Data	Proportion (%)
40k_bangla_newspaper_article	6.93
Bangla novels, story books	2.75
Bangla Wikipedia	5.84
Somewhere-in-Blog post	75.05
Sachalayatan Blog post	9.43

The raw datasets are in pickle format (.pkl) which we aggregated and cleaned for processing. The total number of sentences in the corpus is 39,500,468. We used some filters to remove special characters and English words & characters. We replaced the Bangla numbers in the datasets with "<NUM>" tag as numbers themselves do not carry any meaning. After preprocessing the corpus contains a total of 3,577,910 words. Among these 2,240,897 words are discarded due to single occurrences and our model uses the rest of the 1,337,032 words to create the model. In comparison, in [11], Ismail *et al.* used 521,391 unique words to generate the word embeddings. To our knowledge, our dataset is the largest publicly available Bangla corpus. However, it should be noted that in comparison, Mikolov *et al.* has used 1.6 billion words to train their Skipgram and CBOW models for English [3].

We also checked whether our dataset conserved the Zipfian distribution [22]. The average frequency count of our corpus is 192.26 and the standard deviation is 17,582.45. The most frequent word occurs 6.28% times of the total occurrences while 2nd most frequent word occurs 3.39% and 3rd most frequent word occurs 1.21% times of the total occurrences. Our corpus thus complies to the Zipf's law.

III. Evaluation Methods

In this section we propose datasets for five evaluation methods, namely, analogy prediction, semantic relatedness, synonym detection, antonym detection, and concept categorization for our Bangla word embeddings. Below we describe the datasets and the evaluation methodology in detail.

A. Analogy Prediction

In analogy prediction, based on the semantic relation between two words (*Word1* and *Word2*), we predict a word (*Word4*) that has similar semantic relation with another given test word (*Word3*). The prediction is done using the following equation:

$$Word4 = Word1 + Word3 - Word2 \quad (1)$$

Our evaluation dataset (Analogy_bn) was adopted from Mikolov *et al.* [2] which was translated by two university student volunteers. Our evaluation dataset contains 2,376 combination of Bangla words, from which only 2,102 combinations exist in our vocabulary. Therefore, we evaluate the embedding using these 2,102 combinations. Each row of our dataset is composed of 4 space-separated words where we use the first two words to understand the relation between the words. We predict on the basis of the 3rd word and the 4th word is the given as answer-word that we want to predict. We can use the 4th word for testing the accuracy of our prediction.

Given a pair of words "সুইজারল্যান্ড" (Switzerland) & "সুইস" (Swiss), a predicted word related to a test word "আয়ারল্যান্ড" (Ireland) should be "আইরিশ" (Irish), because "সুইজারল্যান্ড" Switzerland and "সুইস" Swiss have the country-nationality relationship, therefore our third word "আয়ারল্যান্ড" (Ireland) and the predicted word should also conserve the country-nationality relationship. This relationship should be captured in the word embedding vectors when we compute them using equation (1). In other words, if we subtract the vector representation of "সুইজারল্যান্ড" (Switzerland) and that of "সুইস" (Swiss) and add it with the representation of "আয়ারল্যান্ড" (Ireland), we should get a vector having a high cosine similarity with the vector representation of "আইরিশ" (Irish). When we actually compute the vector, we find that the most similar word is not the expected word. But most of the time, the predicted words exist within 5 most similar words. Therefore, we pick 20 most similar words in the vocabulary according to the cosine similarity between those words and choose closest *k* words that are not in the set of given words or not spelling variants of them. We consider a prediction to be correct if the given answer matches any of the *k* chosen predictions. We experimented with different values of *k* ranging from 1 to 20. We calculate the accuracy of the evaluation by checking how many predictions match the given answer.

1. https://github.com/shabdakuhok/Evaluation-datasets-for-Bangla-Word-Embedding
2. https://radimrehurek.com/gensim/models/word2vec.html

B. Semantic Relatedness

Semantic relatedness task evaluates a word embedding based on the degree of semantic similarity between two words. Our semantic relatedness evaluation dataset (Wordsim_bn) was created by translating Wordsim353 dataset [5]. Two university student volunteers did the translation. Our dataset named, comprises of 353 rows each containing 2 words separated by tab and the average of the rating of semantic relatedness assigned by 8 human annotators from two age groups. The first age group consists of 6 undergraduate students with aging from 21 to 25 and the second age group consists of one engineer and one High school Bangla teacher respectively, aging from 40 to 60. To evaluate the embedding, we calculate the cosine similarity as the measure of semantic relatedness between the corresponding vectors of the words pairs. We compute the correlation between the average human- annotated ratings and the computed cosine similarities. We use both Pearson correlation and Spearman rank correlation measures for this purpose.

C. Synonym Detection

This method predicts a word that represents the same semantic meaning as the target-word. We produced the evaluation dataset called "Synonym_bn" from web scraped questions for Bangladesh Civil Service (BCS) examination. The dataset consists of 86 sets of words each containing 6 words separated by commas. The 1st word is the target word. The latter 4 words are the synonym candidates. The last word is the given answer that we want to predict. However, 65 sets of words exist in our vocabulary.

We calculate the cosine similarity between the target word and the 4 candidate words. The word with the largest similarity value is chosen to be the synonym of the target word. For example, we find the cosine similarity of the target word "সূর্য" (Sun) with the 4 given options for synonym, "সুধাংশু" (Moon), "শশাংক" (Moon), "বিধূ" (Moon) and "আদিত্য" (Sun). Here, if the maximum similarity is found with the word "আদিত্য" (Sun) then the prediction is accurate.

D. Antonym Detection

Similar to Synonym detection we also created a dataset for antonym detection. We scraped the web for antonym MCQ questions for our dataset called "Antonym_bn". The dataset contains 172 rows each with six comma-separated words. Like our synonym dataset format, the first word is the target word, 4 word are antonym candidates and the last word is the answer. However, 136 rows of words exist in our vocabulary.

We calculate the cosine similarity between the target word and the four candidate words. The word with the smallest cosine similarity is our predicted word, e.g. 'মুক্ত' (Free) is our target word, and "স্বাধীন" (Independent), "বাহির" (Open), "বদ্ধ" (Closed), "মুক্তি" (Freedom) are our candidates. The minimum similarity is found with the word "বদ্ধ" (Closed), therefore it is selected. We get the accuracy by comparing our predicted word with the given word. If the prediction matches the given answer, we call it a successful prediction.

E. Concept Categorization

In concept categorization, the task is to cluster a set of words into several semantic categories where they should conceptually belong. We created the "Concept_bn" dataset for this purpose. The dataset contains 3 comma separated values where the 1st value is the category ID, second value represents the category, and the third value represents concepts. We have 78 words in our dataset which are clustered into 6 different categories, namely mental state, body action, vehicle, occupation, animal, and change of state. On average there are 16 words in each category. We implemented the k-means clustering algorithm on the "Cluster_bn" dataset. Since there are 6 categories in our dataset we use $k = 6$. The idea is to see whether the word vectors corresponding to the words from the same category fall into the same clusters. Each word belongs to a specific cluster and we determine the ID of the cluster by checking which words of a category occur the maximum number of times in the cluster. We count the words that appear in the correct cluster. We consider a word belonging to a correct cluster if the given Cluster ID of the word and the calculated cluster ID of the cluster are the same.

IV. RESULTS AND DISCUSSION

We proceed to use these 4 different models to perform the evaluation tasks of Analogy Prediction, Semantic Relatedness, Synonym Detection, Antonym Detection and Concept Categorization. The performances of the models are presented in the tables below, where *CB* represents CBOW model, *SG* represents Skipgram model and *d* represents different word embedding dimension sizes. The first observation from our experiments is that, as expected, the performance of the model for the different tasks tends to deteriorate as we reduce the dimension. However, surprisingly, in a number of tasks the performance of 100-dimensional word vectors is similar to that of 300 dimensional vectors. Table II represents the accuracy in percentage of the analogy prediction task. Note, total number of analogy word combinations existing in the dataset is 2,102. We compute the accuracy considering the *k* top similar words.

TABLE II. ACCURACY (%) OF ANALOGY PREDICTION FOR TOP *K* SIMILAR WORD VECTORS FOR DIFFERENT WORD EMBEDDING DIMENSIONS

d	*k*									
	1		2		3		4		5	
	CB	SG	CB	SG	CB	SG	CB	SG	CB	SG
300	22.79	28.12	32.35	40.25	38.44	45.39	43.43	48.76	46.62	51.33
100	18.13	24.50	26.40	33.12	31.26	38.44	35.30	42.58	39.10	45.67
64	18.13	20.23	26.40	28.12	31.26	32.45	35.30	36.63	39.01	39.68
32	7.23	8.37	11.42	12.51	14.93	15.41	16.98	18.51	18.79	21.21

As expected, as we increase *k* the accuracy increases. Instead of looking at the most similar word vector ($k = 1$) if we increase *k* to 5, we see the accuracy increases almost two-folds. For example, we found out that given three analogies, "ব্যাংকক" (Bangkok), "থাইল্যান্ড" (Thailand) and "বেইজিং" (Beijing), the predicted closest word is "ইরান" (Iran) but the correct answer is "চীন" (China), which indeed is in the top k=5 closest words. We have also experiment with different values of *k* greater than 5. At k=7, we achieve 50.81% accuracy. And for k =14 the accuracy rises to 60.13%. After that, however, the rate of increase in accuracy is low. We also observe that Skipgram performs better than CBOW for all dimensions.

Table III shows the results of semantic relatedness. The second row is the Pearson correlation and the third row represents the Spearman correlation.

TABLE III. CORRELATION COEFFICIENTS OF SEMANTIC RELATEDNESS TASK FOR DIFFERENT WORD EMBEDDING DIMENSIONS

d	300		100		64		32	
	CB	SG	CB	SG	CB	SG	CB	SG
Pearson	0.57	0.57	0.57	0.59	0.56	0.59	0.53	0.59
Spearman	0.59	0.62	0.59	0.61	0.58	0.61	0.54	0.59

Referring Table III, we also observe that the performance of CBOW and Skipgram models does not differ too much. Table IV represents the accuracies for Synonym and Antonym detection, where S represents synonym detection and A represents antonym detection task respectively.

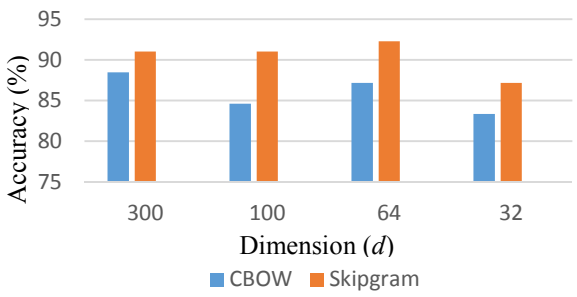
TABLE IV. ACCURACY (%) OF SYNONYM(S) AND ANTONYM (A) DETECTION TASKS FOR DIFFERENT WORD EMBEDDING DIMENSIONS

d	300		100		64		32	
S	CB	SG	CB	SG	CB	SG	CB	SG
	52.31	53.85	55.38	58.46	56.92	50.77	47.69	50.77
A	10.29	5.88	9.56	9.56	8.82	8.82	8.82	17.65

We observe that for Synonym and Antonym evaluations, the models fail when the number of occurrences (count) of the target words in the corpus is low. For example, in case of words such as "কোরক" (Bud, count: 39), "অংস" (Shoulder, count: 26), "গণ্ডদেশ" (Cheek, count: 31) the model fails to predict correctly. We find Skipgram there is no clear winner between Skipgram and CBOW models for these two tasks. Also, we observe higher accuracy for lower dimensional vectors for Skipgram. However, in general, determining synonyms and antonyms using word embeddings is a difficult task given the way distributional hypothesis works. This distributional hypothesis assumes that words that have the same context tend to have a similar meaning. But in natural languages, synonyms and antonyms of a word may appear in the same context, e.g. "রহিম ভালো ছেলে।" (Rahim is a good boy.) and "রহিম খারাপ ছেলে।" (Rahim is a bad boy.). Here "ভালো" (Good) and "খারাপ" (Bad) are antonyms of each other. But they appear in the same context. However, we still find that the model performs better on identifying synonyms than antonyms. This is likely due to the fact that it is more probable for a synonymous word to appear in the same context rather than an antonym.

We perform the k-means clustering algorithm with k = 6. Figure I shows accuracy (%) obtained by the CBOW and Skipgram models with 4 dimension-sizes.

Fig. 1. Accuracy (%) for Concept Categorization for different word embedding dimensions



As mentioned before, we have six concepts. Among them we observe that words belonging to "মানসিক অবস্থা" (mental state) and "শারীরিক ক্রিয়া" (body action) categories get mixed up after performing clustering. For example, words such as "রাগ" (Angry) and "কান্না" can be considered as belonging to both mental state and body action categories. As for the other clusters there is less ambiguity and the performance of models are relatively better as can be observed from the confusion matrix in Table V, for clusters 1 to 6.Table V represents the confusion matrices for concept categorization clusters where *m*, *b*, *v*, *o*, *a* & *c* respectively represent the 6 categories, mental state, body function, vehicle, occupation, animal & change of state.

TABLE V. CONFUSION MATRICES FOR CONCEPT CATEGORIZATION FOR WORD EMBEDDING DIMENSION SIZE 300 USING CBOW AND SKIPGRAM MODELING.

		Predicted					
Actual		*m*	*b*	*v*	*o*	*a*	*c*
	m	10	1	0	0	1	0
	b	0	9	0	0	2	0
	v	0	0	14	0	0	0
	o	0	0	0	14	0	0
	a	0	0	0	0	16	0
	c	1	0	0	0	5	5

CBOW

		Predicted					
Actual		*m*	*b*	*v*	*o*	*a*	*c*
	m	11	1	0	0	0	0
	b	0	11	0	0	0	0
	v	0	0	14	0	0	0
	o	0	0	1	13	0	0
	a	0	0	0	0	16	0
	c	0	5	0	0	0	6

Skipgram

From the confusion matrices in Table V, we observe that for 5 words from 'change of state' clustered with 'body function'category for Skipgram. This is semantically true in Bangla language. But CBOW has clustered the words from 'change of state' category with the words related to 'animal' category, which is nonsensical.

V. CONCLUSION

In this paper, we proposed 5 intrinsic evaluation methods and the datasets for evaluating Bangla word embeddings. We wanted to evaluate the quality of our word embeddings by computing the accuracies obtained by experimenting with these evaluation methods. To our knowledge, our corpus dataset is by far the largest publicly available dataset for Bangla with 1,337,032 unique words. However, our corpus is more skewed towards blog posts with 84.48% of total sentences in the corpus which might be the cause of some performance issues. For analogy prediction, semantic relatedness, and concept categorization the word embedding methods perform moderately well. However, for synonym and antonym prediction accuracy is understandably poor as synonyms and antonyms appear on the same contexts. On the other hand, concept categorization does better than other tasks with average accuracy above 90%. Overall, we observed that Skipgram model performs comparatively better than CBOW model.

In future, we would investigate whether the accuracy of the different tasks can be improved by increasing the size of our corpus and add diversity to it by adding more articles from newspapers, novels, and technical documents. We would also investigate performance of different embedding methods with different context window sizes. We have made the corpus and evaluation datasets publicly available and we believe this will instigate further research in the evaluation of Bangla word embedding.

REFERENCES

[1] Z. S. Harris, "Distributional structure", in Word, 1964, 10(2-3):146–162

[2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corado, and J. Dean, “Distributed representations of words and phrases and their compositionality”, in NIPS, 2013a, pages 3111-3119.

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space”, in Proc. ICLR, 2013.

[4] J. Pennigton, R. Socher, C. D. Manning, “GloVe: Global Vectors for Word Representaion”, in Proc. EMNLP, 2014, doi: 10.3115/v1/D14-1162

[5] Z. Yin and Y. Shen, “On the Dimenstionality of Word Embedding”, in NIPS, 2018.

[6] R. Lebret and R. Collobert, “Word embeddings through Hellinger PCA”, 2014, in EACL, pages 482–490.

[7] P. S. Dhillon, J. Rodu, D. P. Foster, and L. H. Ungar, "Two step CCA: A new spectral method for estimating vector models of words", 2012, in ICML, pages 1551–1558.

[8] P. Li, T. J. Hastie, and K. W. Church, "Very sparse random projections", 2006, in KDD, pages 287–296.

[9] T. Schnabel, I. Labutov, D. Mimmo, and T. Joachims, “Evaluation methods for unsupervised word embeddings”, in ACL, 2015, pages 298-307.

[10] M. Baroni, G. Dinu, and G. Kruzewski, “Don’ Count, predict! a systematic comparison of context-counting vs. contex-predicting semantic vectors”, In ACL, 2014, pages 238-247.

[11] Z. S. Ritu, N. Nowshin, M. M. H. Nahid, and S. Ismail, “Performanc Analysis of Different Word Embedding Models on Bangla Language”, in ICBSLP, Sept. 21-22, 2018, doi: 10.1109/ICBSLP.2018.8554681.

[12] M. Aono and M. Shajalal, “Semantic Textual Similarity in Bengali Text”, in ICBSLP, Sept 21-22, 2018, doi: 10.1109/ICBSLP.2018.8554940.

[13] H. A. Chowdhury, M. A. H. Imon ; M. S. Islam, “A Comparative Analysis of Word Embedding Representations in Authorship Attribution of Bengali Literature”, in ICCIT, Dec 21-23, 2018, doi: 10.1109/ICCITECHN.2018.8631977

[14] A. Ahmad, M. R. Amin, “Bengali word embeddings and it's application in solving document classification problem”, in ICCIT, Dec 18-20, 2016, doi: 10.1109/ICCITECHN.2016.7860236

[15] M. R. Hossain, M. M. Hoque, “Automatic Bengali Document Categorization Based on Word Embedding and Statistical Learning Approaches”, in IC4ME2, Feb 8-9, 2018, doi: 10.1109/IC4ME2.2018.8465632

[16] S. H. Sumit, M. Z. Hossan, T. A. Muntasir, T. Sourov, “Exploring Word Embedding for Bangla Sentiment Analysis”, in ICBSLP, Sept 21-22, 2018, doi: 10.1109/ICBSLP.2018.8554443

[17] M. Al-Amin, M. S. Islam, S. D. Uzzal, “Sentiment analysis of Bengali comments with Word2Vec and sentiment information of words”, in ECCE, Feb 16-18, 2017, doi: 10.1109/ECACE.2017.7912903

[18] M. N. Hasan, S. Bhowmik, M. M. Rahaman, “Multi-label sentence classification using Bengali word embedding model”, in EICT, Dec 7-9, 2017, doi: 10.1109/EICT.2017.8275207

[19] P. V. Veena, M. A.Kumar, K. P. Soman, “An effective way of word-level language identification for code-mixed facebook comments using word-embedding via character-embedding”, in ICACCI, Sept 13-16, 2017, doi: 10.1109/ICACCI.2017.8126062

[20] R. Karim, M.A.M. Islam, S.R. Simanto, S.A. Chowdhury, K. Roy, A.A. Neon, M.S. Hasan, A. Firoze, and R.M. Rahman, "A step towards information extraction: Named entity recognition in Bangla using deep learning", Journal of Intelligent & Fuzzy Systems, vol. Pre-press, no. Pre-press, pp. 1-13, Jul. 2019. doi: 10.3233/JIFS-179349

[21] Z. Shujon, “40k Bangla Newspaper Article”, Internet: https://www.kaggle.com/zshujon/40k-bangla-newspaper-article, Sept. 22, 2018.

[22] G. K. Zipf, “The Psychobiology of Language”, Oxford, England: Houghton, Mifflin, 1935.

Sentimental Style Transfer in Text with Multigenerative Variational Auto-Encoder

Mehedi Hasan Palash
Department of Computer Science and Engineering Shahjalal University of Science and Technology
evonloch@gmail.com

Partha Protim Das
Department of Computer Science and Engineering Shahjalal University of Science and Technology
meparthaprotim@gmail.com

Summit Haque
Department of Computer Science and Engineering Shahjalal University of Science and Technology
summit.haque@gmail.com

***Abstract*—Style transfer is an emerging trend in the fields of deep learning's applications, especially in images and audio data this is proven very useful and sometimes the results are astonishing. Gradually styles of textual data are also being changed in many novel works. This paper focuses on the transfer of the sentimental vibe of a sentence. Given a positive clause, the negative version of that clause or sentence is generated keeping the context same. The opposite is also done with negative sentences. Previously this was a very tough job because the go-to techniques for such tasks such as Recurrent Neural Networks(RNNs) [1] and Long Short-Term Memories(LSTMs) [2] can't perform well with it. But since newer technologies like Generative Adversarial Network(GAN) and Variational Auto-Encoder(VAE) are emerging, this work seem to become more and more possible and effective. In this paper, Multi-Genarative Variational Auto-Encoder is employed to transfer sentiment values. Inspite of working with a small dataset, this model proves to be promising.**

***Index Terms*—text, style, style-transfer, vae, sentiment-transfer**

I. Introduction

In this fast-growing era of deep learning, understanding and transferring of stylistic attributes in numerous fields have been proven very fruitful. Especially in the case of images and other visual forms of data, neural networks have worked wonders in capturing the style elements and manipulating them [3]. This study explores the transfer of styles in linguistic expressions. The possibilities of text style transfer are limitless. It can be used in making customized chatbots that can interact with humans as any human would do. This could come in handy in different organizations and even in personal interests. Moreover, generating parallel data can be another implication of textual style transfer. This is a very important aspect because in many style transfer job the main problem becomes getting the parallel data. Transferring the style from one stream of text to another can solve this problem easily and reliably with much less effort.

Since dealing with textual data is very different from pictorial data, transferring styles is much of a challenge here. Mostly because in transferred images, the slight distortions can be overlooked, which is not the case for linguistic data. Here the model has truly to understand the context and do things according to that. This work is based on a basic stylistic attribute, that is-sentiment. We choose to work with converting a very specific sentiment in human interactions- the positivity and negativity of sentences. We take a sentence or clause with a positive vibe and then change it to a negative one. This is a primitive step towards general style transfer in text. But if this basic problem can be solved, we can hope to get closer to more human-like machine-generated text.

We used Multi Generator Variational Auto-Encoder(VAE) to achieve our goal. We also tried vanilla Auto-Encoder, but this does not work as good as VAEs because they are continuous. On the other hand, both mean and variance are taken into account in VAEs and that simple technique helps get a better result, for this expands the window of opportunity for getting more diverse outputs. Therefore we get more relevant outputs. We use two different decoders to get two types of response. One for positive and another for negative styled output.

II. Related Work

Style transfer with non parallel text has been exercised extensively in recent years. The most influential style transfer work would be of Gatys et al. [3]. It is shown here that the style of images can be transferred. And this and several such works inspire to build some machine similar to them which would work for textual data. Zhang et al. [4] uses CNN for this task, which face many problems mostly because text and image are different in nature and should not be treated alike. Therefore, Neural Machine Translations (NMTs) come into play [5] [6]. Parallel data is used to let the model learn from them. But this is a bit problematic to find parallel texts, and sometimes

978-1-7281-5242-4/19

it is even not possible to get in the post-NMT era of text style transfer, Xu et al. [6], Fu et al. [7], Li et al. [8], Shen et al. [9], Prabhumoye et al. [10] introduce newer techniques like GANs [11] and VAEs to perform this job, in an unsupervised way. That is-without the help of parallel data. There have been many different techniques to evaluate the success of these approaches. Shen et al. [9] uses sentiment modification, word substitution cipher decypherment and word order recovery as human verification factors. They infer styles from a sentences and its original style indicator. A style dependent decoder is used to render them. Moreover, a brilliant technique-cross generated sentences is used to gather more information. Hu et al. [12] employs Variational Auto-Encoders (VAEs) and attribute discriminators to generate sentences, whose attributes are controlled by trained disentangled latent representation of them. Here yelp and amazon datasets are used and evaluated by humans to get more accurate response.

III. Dataset

A. Data Source

We used a dataset on bangla sentences. 4600 comments are collected from prothom-alo [13] news. Every comment has 6 options for tagging. They are 'Surely Negative', 'Slightly Negative', 'Neutral', 'Slightly Positive', 'Surely Positive'. We get the data tagged three times by volunteers for making the tags more reliable. Some examples are shown in figure 1

0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	কিছুটা নেতিবাচক
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	কিছুটা নেতিবাচক
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	কিছুটা নেতিবাচক
3	চুরি নয় লুটপাট।	নিশ্চিত নেতিবাচক
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	নিরপেক্ষ

Fig. 1. Sample data

For the purpose of using the tags appropriately to learn them, we replaced the tags by some numerical values. These are assigned in accordance with their strength. The more the negative a sentence seem to be are assigned more negative values, and the same but opposite goes for the positive sentences. ['Surely Negative', 'Slightly Negative', 'Neutral', 'Slightly Positive', 'Surely Positive'] = [-2,-1,0,1,2]

We then sum up three tags for each instance of the dataset. If the sum is less than zero, we marked it as negative and if greater than zero mark it as positive comment. For example: Suppose a comment has three tags : 'Surely Negative', 'Neutral', 'Slightly Positive'. So the sum for this comment will be : -2 + 1 + 0 = -1. So ultimately we treat this as a negative comment. After eliminating duplicates and unusable sentences, we get 2500 negative comments and 2500 positive comments.

B. Preprocessing

When we work with text data, we have to read stream of characters. Single characters normally don't mean anything. If we combine them together carefully, they will make sense. Tokenizing means split the stream of characters such that we can consider them as semantic unit of language. Tokenization [14] also removes the punctuation marks. Here is an example of tokenization shown in figure 2

```
Sentence:
"রাজপণ্ডিত হব মনে আশা করে । সপ্তশ্লোক ভেটিলাম রাজা গৌড়েশ্বরে ।।"

After tokenization:
['রাজপণ্ডিত', 'হব', 'মনে', 'আশা', 'করে', '।', 'সপ্তশ্লোক', 'ভেটিলাম', 'রাজা', 'গৌড়েশ্বরে', '।', '।']
```

Fig. 2. Tokenizing Sentences

We used Natural Language ToolKit(NLTK) for tokenization. We fixed the sequence length to 20. For sentences smaller than 20, we padded them with a fixed string.

IV. Model Overview

We have used a variational auto-encoder (Doersch et al [15]) with two generators in our model. Normally every variational auto-encoder has two parts. An encoder and a decoder. First, the encoder part maps the input sequence to a latent representation z. Then, the generator part takes samples and makes them saturated with the desired style, which is positive and negative in our case.

We can consider the encoder as a neural network and it takes datapoint x and its output is a hidden representation z. Then, another neural network called generator, samples from z and produces the desired output. z preserves the context of the input and decoder is used for mixing the desired attribute's properties.

In this model depicted in figure 3, two generators are used-one for positive and another for negative. The positive generator is specialized to modify sentences with more positivity. The other is for generating negative sentences. First of all, the input sentence passes through an encoder network. From the encoder network, a parameter of distribution $Q(z|x)$ where x is the vectorized input sentence. The latent vector z is got from the stated distribution. Now this z is the information holder of x. The decoders use z to recreate the sentence with a varied style.

While training, the positive decoder and the negative decoder are separately used so that they can learn about a specific style. That is-when dealing with positive data, we train the negative generator. Again we train the positive generator with the negative data to make it capable of generating positive sentences.

V. Result

We have used human evaluation for determining our model's accuracy. As there is not enough software resource for evaluating style transfer in Bangla. Human evaluation has been used in extensively in the validation of recent linguistic tasks such as in the works of Shen et al [9].

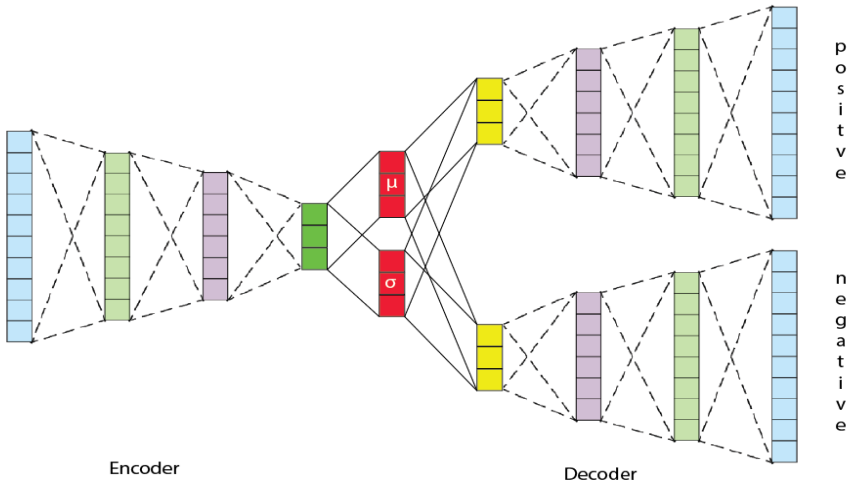


Fig. 3. High Level view of the model

We have measuerd the following parameters for determining the success of an output:

- ***Grammatical Correctness*** This metric is used to see how much the model can understand the structure of the language.
- ***Context Similarity*** Whether the output could capture the context is expressed as this.
- ***Polarity*** This is the most important point in our task- whether the output convey positive or negative vibe.

We have copied the input of the model and used it as the output to get a baseline result. Since human comments are used as input baseline output, the output has hundred percent grammatical correctness and context similarity with input. But it has not achieved the correct polarity because the negativity of a positive sentence is small, and vice versa.

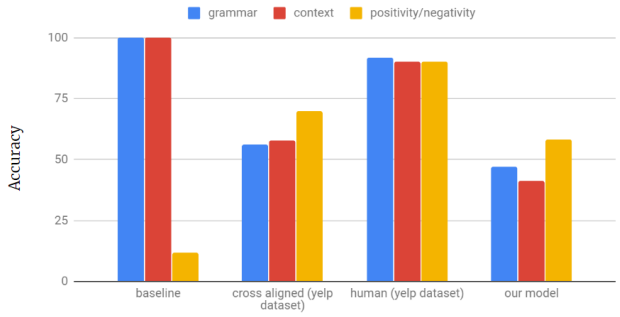


Fig. 4. Comparing our results with state of the art models

Comparing with the human tested results with the yelp dataset, all the parameters(grammar, context, positivity/negativity) are much higher than all the state of the art models. This means that machines couldn't be able to beat humans in this respect. But among the best performing models, cross aligned technique is proven to be the best. Since our dataset is eerily small and in bangla, we couldn't achieve such greateness. Nonetheless, we have got closer to that model in grammar and sentiment transferability, though we need to focus on the fact that context-wise we are far from expectation.

VI. Analysis

The sentiment transferability and grammatical correctness of our model are not so far from the model prescribed by Shen et al.(2017) [9]. But the context preservation is not satisfactory. The dataset we have used has variety of contexts. Some are about politics, some are about sports and so on. But Shen et al.(2017) [9] used a standard dataset from amazon product review. So, the context preservation of our model is not as good as their model. On the other hand if we could use data from a single context then it might have performed better.

A sample output generated by our model is shown here in figure 5. Here, we can see that for the given positive sentence,

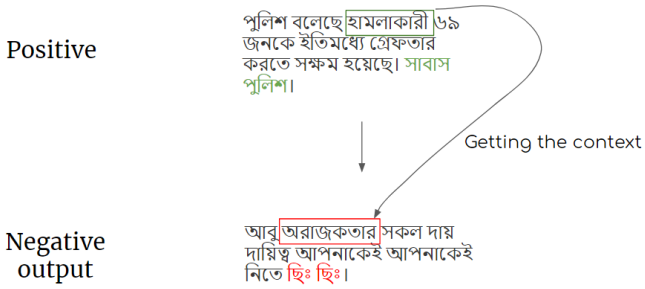


Fig. 5. Positive to generated Negative output

our model tried to get the context(green boxed) and then staying within the context, it tries to generate totally opposite a sentence. Similarly, if a negative sentence is fed into the machine as in figure , the context is still captured and then the

style of this negativity is changed, and it becomes a positive sentence.

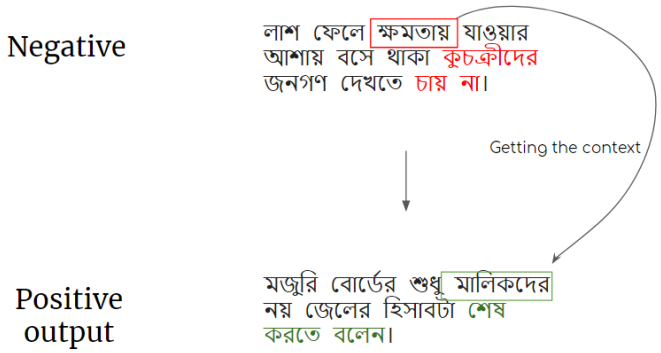


Fig. 6. Negative to generated Positive output

VII. Conclusion

A. Discussion

Transferring style in text is not much explored field. It's still a newer area than other explorable fields like images and audios. Yet it is so much promising. Because there are so many things we can do with textual data. With the power to generate new sentences is alone a huge task to do with a machine. And if we can apply our customized styles and personalization to it, this sounds even more wonderful, because up until now this was solely a humane capability.

We explore this vast field's only a small portion-translating negative sentences into positive ones and translating positive sentences into negative ones. Our model is able to succeed in some cases, the accuracy we get is 53.2%. This might not be very good result, but our model couldn't do much well because of shortage of data. If we could get hold of much larger dataset, this model could achieve more, mostly because our model is very much data dependent.

B. Future Work

Working with sentimental values is an initial step in textual style transfer process. In this paper only the positivity and negativity of a sentence are exploited. There are many other aspects of styles in a sentence that can be explored in this manner, like-gender, tense, political stand etc. If these basic style-bearing aspects can properly be understood and make the machine understand, it will have many interesting applications, like-personalized chatbots and even making an universal language translator.

VIII. Acknowledgement

We would like to thank the NLP group of Shahjalal University of Science and Technology for the necessary insight and expertise that greatly assisted the research.

References

[1] A. Karpathy, "The Unreasonable Effectiveness of Recurrent Neural Networks," May 2015. [Online]. Available: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[2] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2414–2423.

[4] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.

[5] J. M. Hughes, N. J. Foti, D. C. Krakauer, and D. N. Rockmore, "Quantitative patterns of stylistic influence in the evolution of literature," *Proceedings of the National Academy of Sciences*, vol. 109, no. 20, pp. 7682–7686, 2012.

[6] W. Xu, A. Ritter, B. Dolan, R. Grishman, and C. Cherry, "Paraphrasing for style," in *Proceedings of COLING 2012*, 2012, pp. 2899–2914.

[7] Z. Fu, X. Tan, N. Peng, D. Zhao, and R. Yan, "Style transfer in text: Exploration and evaluation," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[8] J. Li, R. Jia, H. He, and P. Liang, "Delete, retrieve, generate: A simple approach to sentiment and style transfer," *arXiv preprint arXiv:1804.06437*, 2018.

[9] T. Shen, T. Lei, R. Barzilay, and T. Jaakkola, "Style transfer from non-parallel text by cross-alignment," in *Advances in neural information processing systems*, 2017, pp. 6830–6841.

[10] S. Prabhumoye, Y. Tsvetkov, R. Salakhutdinov, and A. W. Black, "Style transfer through back-translation," *arXiv preprint arXiv:1804.09000*, 2018.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[12] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing, "Toward controlled generation of text," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1587–1596.

[13] "Prothom Alo," https://www.prothomalo.com/, accessed: 2019-07-31.

[14] "Tokenization," https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html , accessed: 2019-07-10.

[15] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

International Conference on Bangla Speech and Language Processing(ICBSLP), 27-28 September, 2019

Bengali Question Answering System for Factoid Questions: a statistical approach

Sourav Sarker
Computer Science and Engineering
Shahjalal University of
Science and Technology
Sylhet-3114, Bangladesh
sourav39@student.sust.edu

Syeda Tamanna Alam Monisha
Computer Science and Engineering
Shahjalal University of
Science and Technology
Sylhet-3114, Bangladesh
alammonisha@gmail.com

Md Mahadi Hasan Nahid
Computer Science and Engineering
Shahjalal University of
Science and Technology
Sylhet-3114, Bangladesh
nahid-cse@sust.edu

***Abstract*— Question answering system in recent days is one of the most trending and interesting topics of research in computational linguistics. Bengali being among the most spoken languages in the world has yet faced difficulties in computational linguistics. This paper demonstrates an attempt to develop a closed domain factoid question answering system for Bengali language. Our proposed system combining multiple sources for answer extraction extracts the answer having the accuracy 66.2% and 56.8% with and without mentioning the object name respectively. The system also hits around 72% documents from which the answer can be extracted. Besides the sub-parts of our system, the question and document classifier provides 90.6% and 75.3% accuracy respectively over five coarse-grained categories.**

***Index Terms*— Question Answering (QA) System, Bengali Question Answering System, Factoid QA System**

I. INTRODUCTION

An automated question answering system is a program that is able to converse with the user in natural language in such a way that no one is able to differentiate it from a real human being. In today's time question answering system is one of the hot topics in Natural Language Processing (NLP) research. It can be either closed-domain based or open-domain based. Closed-domain question answering deals with questions under a specific domain whereas open-domain question answering deals with questions about nearly anything based on world knowledge. In general a question is of two types: factoid question which is satisfied by a short text and descriptive or complex question which needs to be answered in more than one line.

Being one of the hot topics the works being done on question answering system is increasing day by day. In English a lot of works have been done on question answering system and also there exists a number of working question answering systems. But although Bengali being one of the most widely spoken languages the works done for Bengali language compared to English is yet very low.

We worked towards developing a question answering system for only factoid questions for Bengali language on a closed domain. Initially we have chosen Shahjalal University of Science & Technology (SUST) as our domain because every year during the admission test of Shahjalal University of Science & Technology (SUST) the candidates have a lot of queries related to SUST. In general there are different social media groups which are formed due to helping the candidates with information, updates, queries etc. Candidates also ask questions in the official website of admission test. So we wanted to create a common platform for the candidates where they can get answers to the queries. Thus we wanted to build a Bengali question answering system which can reply to the queries instantly.

II. RELATED WORKS

A lot of researches have been done and are also ongoing on question answering system in different languages. Besides there are also works on question classification, question features, question taxonomies and answer extraction which are the sub-parts of question answering system. There have been number of question answering systems developed since the 1960s. Among the earlier question answering systems some of them are domain restricted and some are generalized.

AnswerBus is an open-domain question answering system where the information related to the answer is retrieved from the web in sentence level. The authors used five search engines (Google, Yahoo, Altavista, WiseNut and Yahoo News) to extract the web documents which contain the answers to the questions of the users. The current rate of correct answers to TREC-8's (Text REtrieval Conference-8) 200 questions is 70.5% [1]. JAVELIN is

978-1-7281-5242-4/19

another open-domain based question answering system [2]. The team suggested three QA runs JAVELIN I, JAVELIN II [3] and JAVELIN III [4] among which JAVELIN III can be used for cross-lingual task. Some of the earlier domain restricted QA systems are BASEBALL and LUNAR [5][6]. A system was described by author's Abney et. al., 2000 that handles arbitrary questions by producing a candidate list of answers ranked by their plausibility [7]. The system was evaluated on the TREC question-answering track which showed that the correct answer to queries appeared in the top five answers 46% of the time with a mean score of 0.356.

Nowadays there is a remarkable increase in the research for Bengali language in the field of Natural Language Processing (NLP). Question answering system being one of the hot topics the research works for question answering system for Bengali language is also increasing at this time.

Author's Banerjee et al., 2014 made the first attempt on building a factoid question answering system for Bengali language where the answer is processed by help of named entities [8]. They also discussed the challenges faced to develop the system for Bengali language. A question answering system was developed for Bengali using anaphora-cataphora resolution [9]. Authors experimented the system for both Bengali and English language and they used semantic and syntactical analysis. The model reduces the complexity of using noun instead of pronoun for the requested answer with respect to the given question queries for Bengali and provides 60% accuracy.

III. CORPUS CONSTRUCTION & ANALYSIS

Dataset is an important issue towards the development of a question answering system. To build a question answering system we need two types of data. One is the knowledge base and the other is question database. Then the data are prepared individually for the training of question classification and document categorization.

As we have worked on a closed domain based Bengali factoid question answering system on the domain Shahjalal University of Science & Technology (SUST) we could not get any prepared dataset for our work. So we had to prepare our own questions and knowledge base. This data collection task was one of the challenging tasks as we had no resource available. We used different sources for our data collection part. We had to collect data both for questions and documents in different ways.

A. Question Database

Various sources were used for building the question dataset such as crowd sourcing, social media, manual generation etc. But among them, we collected most of the data from crowd sourcing. We collected questions from the students of Shahjalal University of Science & Technology (SUST) as well as from the official website of SUST where we got the frequently asked questions by the candidates. We also prepared questions from different documents, articles and news on Shahjalal University of Science & Technology (SUST).

B. Knowledge Base

Knowledge base is the set of documents from where the answers of the questions are to be searched and extracted. We have built our knowledge base based on the website which carries the information solely about Shahjalal University of Science & Technology (SUST) like `www.sust.edu`, `en.wikipedia.org/wiki/Shahjalal_University_of_Science_and_Technology` etc. and news from different web portals. We also collected some of the paragraphs about SUST by crowd-sourcing.

TABLE I
Sources of Question Database and Knowledge Base

Data Type	Collection Type	Source	Amount of Data
Question	Crowd Sourcing	2nd Year students, Dept of CSE,SUST	11300
	Social Media Data Crawling	Facebook Group: SUST Admission Aid	1055
	Manual Generation	Authors	3000
Document	Crowd Sourcing	2nd Year students, Dept of CSE,SUST	40
	SUST based websites	www.sust.edu, Wikipedia sust	100
	News Portals	Bdnews24.com, eprothom-alo.com, www.sustnews24.com thedailystar.com etc.	80

IV. ANSWER TYPE TAXONOMY

As we have worked for a closed domain question answering system on the domain Shahjalal University of Science & Technology (SUST) we defined five coarse-grained classes related to SUST for question classification and document categorization. The questions and documents are classified in these five categories. Table II shows the category details.

TABLE II
Question and Document Categories

Class Name	Short form	Description
Administration	ADS	Questions that require administrative information as answer and documents of administrative type are of ADS class
Admission	ADM	Questions that require admission related information as answer and documents of admission type are of ADM class
Academic	ACD	Questions that require academic information as answer and documents of academic type are of ACD class
Campus	CAM	Questions that require campus related information as answer and documents related to campus are of CAM class
Miscellaneous	MISC	Questions that require any information other than the above four types and documents other than the four types are of MISC class

TABLE III

Sample Tagged Question

Question	Label
সাস্টে মোট কতটি সিট আছে	ADM
সাস্টে প্রতিবছর কতটি করে স্কলারশিপ দেয়	ACD
সাস্ট মানে কি	MISC
সরকারী বৃত্তি প্রাপ্তদের কি ভর্তি ফি দিতে হয়	ADS
সাস্ট ক্যাম্পাসে কি খাবারের দাম সহনীয়	CAM

TABLE IV

Sample Tagged Document

Document	Label
মানবতার জন্য শেখো স্লোগানকে প্রতিপাদ্য করে ২০১২ সালের ১১ জানুয়ারী কিছু সচেতন শিক্ষক-শিক্ষার্থীর হাত ধরে শুরু হয় শাহজালাল বিজ্ঞান ও প্রযুক্তি বিশ্ববিদ্যালয়ের একমাত্র প্রকৃতি ও পরিবেশ বিষয়ক সংগঠন গ্রিন এক্সপ্লোর সোসাইটির পথচলা। বিশ্ববিদ্যালয়ের শিক্ষার্থীদের পাশাপাশি তৃনমূল পর্যায়ে পরিবেশ সচেতনতা বৃদ্ধিতে কাজ করে যাচ্ছে সংগঠনটি।	CAM

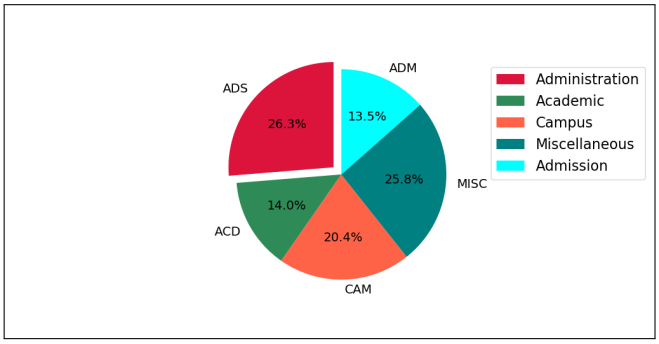


Fig. 1. Percentage of questions in each category

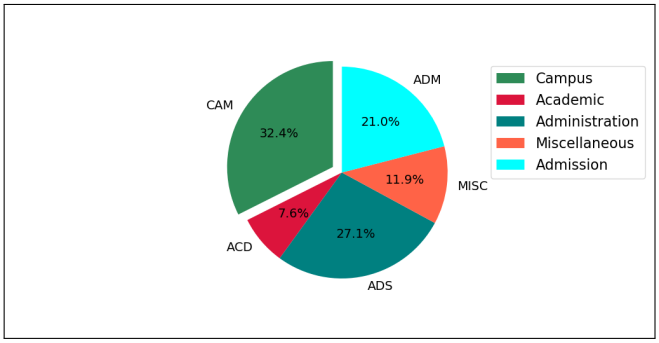


Fig. 2. Percentage of documents in each category

V. METHODOLOGY

The architecture that we have proposed for our system is shown in figure 3. As shown in the figure we have used three sources for answer extraction: mapped question, collection of documents and internet resource.

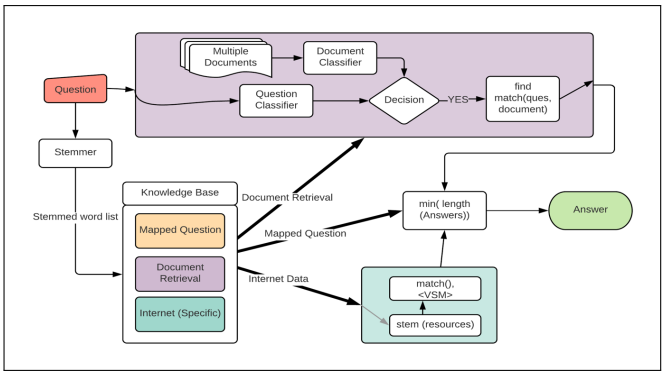


Fig. 3. Proposed architecture for our system

As depicted in the figure for extracting an answer at first a list of stem words is created from the question. We have used a Bengali rule based stemmer for the process. At first the answer is looked for in the frequently asked section. Here we have mapped the most frequently asked questions to the answers. If the answer is found here considering common tag words (শাবি, শাবিপ্রবি, শাহজালাল বিশ্ববিদ্যালয়, শাহজালাল বিজ্ঞান ও প্রযুক্তি বিশ্ববিদ্যালয়, সাস্ট) then an answer will be provided otherwise the answer will be searched in the categorized documents. In this case the asked question is first classified to the expected answer type and then searched in the documents of the same type and if the answer is not found in the categorized documents then it searches in the internet sources which are predefined for the particular domain and provides an answer.

We wanted to build a question answering system for Bengali language on a closed domain and not having enough resources to build it, we had to work from the scratch. We have divided our methodology to answer a factoid question in four basic parts:

1) Data Preparation
2) Question Classification
3) Document Categorization
4) Answer Processing

A. Data Processing

As we collected data from different sources we needed to prepare and clean the dataset. The following tasks were done to prepare the corpus for further processing.

1) Removed stop words (অবশ্য, অনেক, অনেকে, অনেকেই, অন্তত, ভাবে, মধ্যে etc.)
2) Removed sign characters
3) Tokenized the words in special case
4) Checked and made correction of spelling mistakes of raw data manually
5) Rechecked the labels that were assigned manually

B. Question Classification

Question classification is an important first step towards building a question answering system [10]. The classification of a question to expected answer type reduces the search space by a considerable amount [11]. We have used four machine learning algorithms Stochastic Gradient Descent(SGD), Decision Tree(DT), Support Vector Machine(SVM) and Naive Bayes(NB) for our question classification phase [12]. For feature extraction similar words and both bi-gram and tri-gram were used where tri-gram provides better results. Again, we have tested dynamic word clustering model as deep learning methods are getting popularity in these days [13].

C. Document Categorization

The documents that we have collected were also classified to predefined categories. For document categorization we have used a different approach other than question classification. As document classification follows passage classification we used word embedding here. Word embedding is one kind of learned representation for text where words that have the same meaning have a nearly same representation. We have used fastText as our embedding technique and implemented convolutional neural network(CNN) classifier.

D. Answer Processing

The most important and challenging part of our question answering system is the answer extraction method. As shown in figure 3 we have used three sources for answer extraction process. The following techniques have been used for our answer extraction phase.

1) Vector Space Model (VSM): In vector space model documents and queries are represented as vectors of features representing the terms that occur within the collection. A document d_j and a question q can be represented by the following vectors.

$$\vec{d_j} = (w_{1,j}, w_{2,j}, w_{3,j}, w_{4,j}, \ldots, w_{n,j})$$

$$\vec{q} = (w_{1,q}, w_{2,q}, w_{3,q}, w_{4,q} \ldots, w_{n,q})$$

Here the number of dimensions in the vector is the total number of terms in the whole collection. The match found by the question from a particular document is measured by the following similarity function:

$$sim(\vec{q}, \vec{d_j}) = \frac{\sum_{i=i}^{N} w_{i,q} * w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,q}^2} * \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}$$

After calculating this score an answer is generated using the sentence having the highest rank.

2) Comparison of VSM and Edit distance: We have used a comparison between edit distance and VSM and received the maximum value between edit distance and VSM matching. Comparing both, the answer with the highest score is provided and if multiple answers have the same score then the one with minimum length is selected as factoid questions require single facts as answer.

E. Performance Metric

We have used the accuracy measure to evaluate our experiments. Accuracy is the number of correct predictions made by a model over all kinds of predictions made which is measured by the formula:

$$Accuracy = \frac{TP + TN}{P + N}$$

where,

P = total number of positive samples

N = total number of negative samples

TP (True Positive) = number of cases when the actual class of the sample is true and also predicted as true

TN (True Negative) = number of cases when the actual class of the sample is false and also predicted as false

VI. EXPERIMENTS & RESULT ANALYSIS

For the training purpose of question classification and document categorization we had to manually tag the questions and documents. In both cases we have used 75% of the dataset for training purpose and 25% for testing. For question classification Support Vector Machine(SVM) with linear kernel provides the best accuracy 90.6% among classifiers Stochastic Gradient Descent(SGD), Decision Tree(DT), Support Vector Machine(SVM) and Naive Bayes(NB) that we used [12].

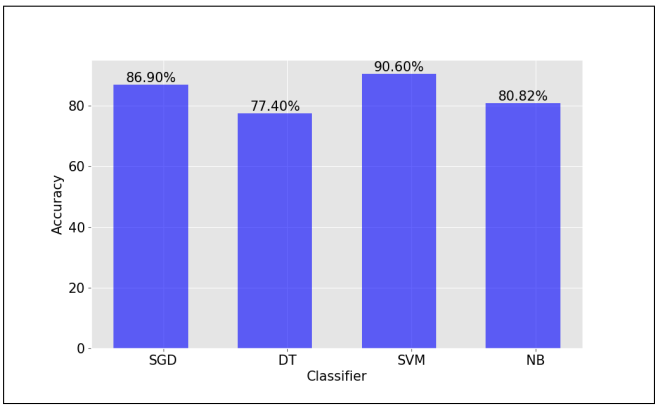


Fig. 4. Best performance for different classifier for question classification

For document categorization we have implemented convolutional neural network(CNN) using fastText skip-gram word embedding technique. Accuracy of 75.3% was obtained for document categorization over the five

categories that we defined.

Following our answer extraction technique the proposed system provides around 56.8% accuracy without mentioning the object name like শাবি, শাবিপ্রবি, শাহজালাল বিশ্ববিদ্যালয়, শাহজালাল বিজ্ঞান ও প্রযুক্তি বিশ্ববিদ্যালয়, সাস্ট and around 66.2% with mentioning the object name. We have received a document hit of around 72% for our system.

TABLE V
Sample answers to asked questions

Question	Extracted Answer
সাস্টে কয়টি অনুষদ রয়েছে	৭ টি
সাস্টের বর্তমান ভিসির নাম কি	অধ্যাপক ফরিদ উদ্দিন আহমেদ
ছেলেদের হল কয়টি	৩ টি

TABLE VI
Performance of overall system

Type	Score	Condition
Document Hits	72%	Considering object name
Answer accuracy	66.2%	Considering object name
Answer accuracy	56.8%	Without mentioning object name

VII. CONCLUSION

In our whole time of working with Bengali question answering system we have tried to build a generic factoid question answering system that is if we just provide the system with the knowledge base and question database, it will be able to extract answers from them. We have built a corpus consisting of 15355 questions and 220 documents on our domain Shahjalal University of Science & Technology for our system. The dataset or corpus creates the main hazard for building a question answering system or any language tool. No well-established Bengali language tool has been released till date. So for the better performance of a question answering system natural language processing tools like named entity recognizer, parts of speech tagger, stemmer etc. and also the corpus which is base of the question answering system need to be developed.

So far we have worked with factoid questions only. We have future plans to forward the process by taking complex and descriptive questions into consideration. We also want to implement more techniques to enhance the performance of the system.

References

[1] Z. Zheng, "Answerbus question answering system," in *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., 2002, pp. 399–404.

[2] E. Nyberg, T. Mitamura, J. G. Carbonell, J. P. Callan, and K. Collins-Thompson, "The javelin question-answering system at trec 2002," 2002.

[3] E. Nyberg, R. E. Frederking, T. Mitamura, M. W. Bilotti, K. Hannan, L. Hiyakumoto, J. Ko, F. Lin, L. V. Lita, V. Pedro *et al.*, "Javelin i and ii systems at trec 2005." in *TREC*, vol. 2, no. 1, 2005, p. 1.

[4] T. Mitamura, F. Lin, H. Shima, M. Wang, J. Ko, J. Betteridge, M. W. Bilotti, A. H. Schlaikjer, and E. Nyberg, "Javelin iii: Cross-lingual question answering from japanese and chinese documents." in *NTCIR*, 2007.

[5] B. F. Green Jr, A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball: an automatic question-answerer," in *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*. ACM, 1961, pp. 219–224.

[6] W. A. Woods, "Progress in natural language understanding: an application to lunar geology," in *Proceedings of the June 4-8, 1973, national computer conference and exposition*. ACM, 1973, pp. 441–450.

[7] S. Abney, M. Collins, and A. Singhal, "Answer extraction," in *Proceedings of the sixth conference on Applied natural language processing*. Association for Computational Linguistics, 2000, pp. 296–301.

[8] S. Banerjee, S. K. Naskar, and S. Bandyopadhyay, "Bfqa: A bengali factoid question answering system," in *International Conference on Text, Speech, and Dialogue*. Springer, 2014, pp. 217–224.

[9] S. Khan, K. T. Kubra, and M. M. H. Nahid, "Improving answer extraction for bangali q/a system using anaphora-cataphora resolution," in *2018 International Conference on Innovation in Engineering and Technology (ICIET)*. IEEE, 2018, pp. 1–6.

[10] S. Banerjee and S. Bandyopadhyay, "Bengali question classification: Towards developing qa system," in *Proceedings of the 3rd Workshop on South and Southeast Asian Natural Language Processing*, 2012, pp. 25–40.

[11] E. Haihong, Y. Hu, M. Song, Z. Ou, and X. Wang, "Research and implementation of question classification model in q&a system," in *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, 2017, pp. 372–384.

[12] S. T. A. Monisha, S. Sarker, and M. M. H. Nahid, "Classification of bengali questions towards a factoid question answering system," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT 2019)*. IEEE, 2019, pp. 660–664.

[13] Z. S. Ritu, N. Nowshin, M. M. H. Nahid, and S. Ismail, "Performance analysis of different word embedding models on bangla language," in *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*. IEEE, 2018, pp. 1–5.

Does Word2Vec encode human perception of similarity? A study in Bangla

Manjira Sinha
Center for Education Technology
IIT Kharagpur
Kharagpur, India
manjira87@gmail.com

Rakesh Dutta
Dept. of Computer Science and Application
University of North Bengal
Siliguri, India
rakeshhijli@gmail.com

Tirthankar Dasgupta
Innovation Lab
Tata Consultancy Services,
Kolkata, India
iamtirthankar@gmail.com

***Abstract*—The quest to understand how language and concepts are organized in human mind is a never-ending pursuit undertaken by researchers in computational psycholinguistics; simultaneously, on the other hand, researchers have tried to quantitatively model the semantic space from written corpora and discourses through different computational approaches - while both of these interacts with each other in-terms of understanding human processing through computational linguistics and enhancing NLP methods from the insights, it has seldom been systematically studied if the two corroborates each other. In this paper, we have explored how and if the standard word embedding based semantic representation models represent the human mental lexicon. Towards that, We have conducted a semantic priming experiment to capture the psycholinguistics aspects and compared the results with a distributional word-embedding model: Bangla word2Vec. Analysis of reaction time indicates that corpus-based semantic similarity measures do not reflect the true nature of mental representation and processing of words. To the best of our knowledge this is first of a kind study in any language especially Bangla.**

***Index Terms*—Computational Psycholinguistics, Semantic Priming, Mental Lexicon, Response Time, Degree of Priming, Word2vec model.**

I. Introduction

The representation of words in the human mind is often termed as Mental lexicon (ML) [1]–[3]. Although, words have various degree of association that are governed by different linguistic and cognitive constraints, the precise nature of the interconnection in the mental lexicon is not clear and remains a subject of debate over the past few decades. However, a clear understanding of how words are represented and processed at the mental lexicon will not only help enhance our knowledge of the inherent cognitive processing but may also be used to develop cognitively aware Natural Language Processing(NLP) applications.

A lot of cognitive experiments are being carried out in different languages to study the semantic representation of words in the ML. Typically, semantic priming experiments are used to perform such studies [4], [5]. Priming involves exposure of a stimulus (called prime) that results in quicker recognition of a related stimulus (called the Target). For example, the amount of time required to recognize a word **BRANCH** will be small if it is preceded by a related stimulus like **TREE** as compared to an unrelated stimulus like **HOUSE**. The data collected from such cognitive experiments are then used to develop robust computational models of representation and processing of words in the ML.

Attempts have also been made to provide computational models of representation of semantically similar words in the mental lexicon. Among them, one of the most commonly used is the distributional semantic model of representation. The idea behind distributional semantics is that words with similar meanings are used in similar contexts [6]. Thus, semantic relatedness can be measured by looking at the similarity between word co-occurrence patterns in text corpora. In linguistics, this idea has been a useful line of research for the past couple of decades [7], [8]. Recently, due to the advancements in the area of deep neural networks, a more robust form of distributional semantics models have been proposed in the way of word2vec word embedding. However, most of these works are based on languages like English, French, German, Arabic, and Italian. Very few attempts have been made to perform such studies for Bangla ML. Moreover, an important aspect of a study is to verify the fact of whether the distributional approach of words representation is truly the way human mental lexicon works.

In this research, we have explored how and if the standard word embedding based semantic representation models represent the human mental lexicon. The paper discusses various cases where it has been shown that highly similar words identified by the computational models seldom shows any priming effect by the users. On the other hand, word pairs having low corpus similarity may result in a high degree of priming.

The objectives of the paper are as follows:

- We conduct a semantic priming experiment over a set of 300-word pairs to study the organization and representation of Bangla words in the mental lexicon.
- For each of the 300 Bangla word pairs, we have computed their word embedding using the word2vec

technique and computed the semantic distance between each of the word pairs.
- We try to observe whether there exist any correlation between the computed similarity score and the priming effect. Here, the null hypothesis is words having high cosine similarity scores will show a high degree of priming.

II. Semantic Priming Experiment on Bangla Semantically Similar Words

In order to study the effect of priming on semantically related words in Bangla, we have execute the masked priming experiment as discussed in [9]–[11]. In this technique, the prime word (say *chor* (thief)) is placed between a forward pattern mask and the target stimulus (say *polisa* (Police)), which acts as a backward mask. This is illustrated below.

$$prime(1000ms)chora(THIEF) \rightarrow target(2000ms)pulisa(POLICE)$$

Once the target porbe is presented to the participants for a given period of time, they are asked to decide whether the given target word is valid or not. The participants enter their decision by pressing the key 'J' (for valid words) and 'K' (for invalid words) of a standard QWERTY keyboard. The time taken to press any one of the key after the display of the target word (also called the response time (RT)), is recorded by the system timer. We display the same target word once again but with a different visual probe called the CONTROL word. The CONTROL word do not show any semantic, or orthographic similarity with either the prime or the target word. For example, *baYaska (aged)* and *briddha (aged)* is a *prime-target* pair, and the corresponding control-target pair could be *naYana (eye)* and *briddha (aged)*. We use the DMDX software tool[1] to conduct all the experiments conducted in this work.

A. Data and Experiment

We choose 300 word pairs from a Bangla corpus of around 3.2 million unique words [2]. The corpus consist of the literary works of famous Bangla authors; we have also extended the corpus by adding texts from Bangla Wikipedia, News sources, and Blogs. The words are chosen is such a way that they represents a substantial amount of distribution over the entire corpus. This will further be useful to construct the word embedding for computing the semantic similarity.

B. Implicit Perception of Semantic Similarity

1) Methods and Material: There were 300 prime-target pairs classified into two different classes. For each of the targets a *prime* and a *control* word have been chosen. Class-I primes have high degree of relatedness (e.g. **সূর্য**(Surya(sun)) – **অস্ত**(asta(sunset))), where as class-II primes have a low degree of relatedness (**ছাগল**(Chagal(goat)) – **অস্ত**(asta(sunset))).

The controls, do not possess any semantic, orthographic or morphological relationship with the target word. It is important to restrict the subject to make any strategical guess regarding the relation between pairs of words. Thus, the prime-target and the control-target words were mixed with equal number of fillers, which are out of vocabulary words such as non-words.

2) Participants: The experiments were conducted on 100 native Bangla speakers with at least a graduation degree. The age of the subjects varies between 20 to 30 years.

3) Result: Extreme reaction time and incorrect responses of the RT in the lexical decision (about 7.5%) were not included in the latency analysis. We set extreme reaction time for one subject as the median lexical latency of that essence subject. Table I depicts the average reaction time (RT) of some prime-target and control-target word pairs.

Table I
The average reaction time (RT) of the prime-target and control-target word pairs.

Prime Word	*Target Word*	*Control Word*	*Average RT(P-T)*	*Average RT(C-T)*
চোর	ডাকাত	জল	548.62	786.89
লোভ	লোভী	ভয়	636.71	716.80
রোগ	রোগী	বাতাস	700.14	821.81
জল	বায়ু	বই	612.81	726.96
বিচার	বিচারক	জামা	669.99	809.42

Then we calculate the degree of priming (DOP) i.e. the average reaction time (control-target) minus the average reaction time (prime-target) words pairs for each word pair. This is represented as follows:

$$Degreeof Priming(DOP) = Diff(RT(C_T), RT(P_T)) \quad (1)$$

After getting the result of DOP, we again calculate the average of DOP across all users.

III. Distributional Approach towards Measuring Semantic Similarity

Recently, word embedding techniques proposed by Mikolov and et al. (2013a) [12] argued that neural network based word embedding (word2vec) models (see Figure 1) are efficient at creating robust semantic spaces. Typically word embeddings are computed by using two techniques: a) Skip Gram model and b) Continuous Bag of Word model (CBOW). The skip gram model in one hand, considers a central word and tries to predict the context words that best fits the target word. On the other hand, the CBOW model tries to predict a target word given the context words. Consider for example the sentence: **কচ্ছপ একধরনের সরীসৃপ যারা পানি এবং ডাঙা দুই জায়গাতেই বাস করে।**

Let **কচ্ছপ** be the input to the proposed neural network. Our objective here is to predict a target word **সরীসৃপ** using

[1] http://www.u.arizona.edu/ kforster/dmdx/download.htm
[2] obtained from www.snltr.org

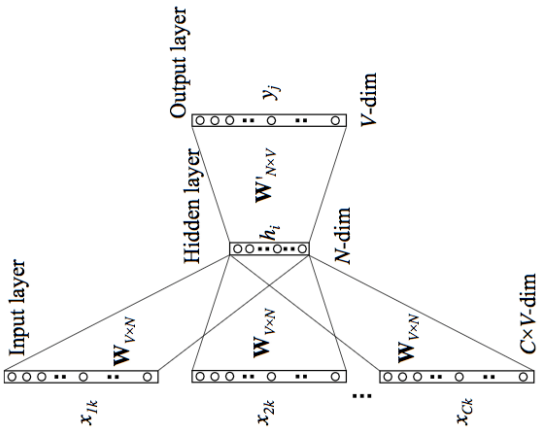


Figure 1. Illustration of the Word2Vec Model

the single context input word **কচ্ছপ**. We use the one hot encoding of the input word and measure the output error compared to one hot encoding of the target word. Thus, the vector representation of the target word is learned by learning to predict the same target word. The basic architecture of the CBOW model is depicted in Figure 1.

We have computed the word2vec embedding of each of the words present in the 300 prime-target, control-target pairs. We use some pre-trained word2vec models i.e model-2 [3] and model-3 [4], and also trained word2vec and generated embedding with different dimension on the corpus discussed in section II-A (model 1). From the different sets of embedding, we have computed the semantic similarity between each of the word pairs using the cosine similarity measure:

$$Cos(w_i, w_j) = \frac{\bar{w_i}.\bar{w_j}}{|w_i||w_j|} \tag{2}$$

Next, we compared the scores with the DOP collected from the priming experiment. Table II depicts the correlation between the different embedding models and the reaction times obtained from the priming experiment.

Table II
Comparison table with different models with respect to the human annotated Data.

Model name	***VOCAB***	***Dimension***	***Co-relation***
Model 1	427261	300	-0.2422
Model 1	427261	400	-0.2104
Model 2	10059	200	-0.0994
Model 3	145350	300	0.0203

From Table II we observed that there is a very week correlation between the Word2Vec models and the Reaction Time data.

[3]https://github.com/Kyubyong/wordvectors
[4]https://fasttext.cc/docs/en/crawl-vectors.html

Table III
Similarity score of Human annotated (DOP) vs Word2vec annotated.

Word1	***Word2***	***Control Word***	***Degree of priming***	***Word2Vec***
চোর	ডাকাত	জল	238.28	0.3236
দেশ	দেশী	জানালা	194.61	0.2549
বিদ্যালয়	ছাএ	পাখি	127.10	0.3771
উদ্দেশ্য	বিধেয়	সাঁতার	309.81	0.2678
ইতিহাস	ভূগোল	বালক	197.67	0.4535
রাম	রাবণ	রাঁধুনি	160.19	0.4060
তাঁতী	কাপড়	রবি	153.19	0.3686
ভোর	আলো	মাঝি	164.45	0.3758
বিদ্যুৎ	গর্জন	দয়া	131.00	0.2584
পিয়ন	চিঠি	চাঁদ	166.10	0.3292
রাত	স্বপ্ন	ডানা	189.92	0.2880
মধুসূদন	রবীন্দ্রনাথ	মাথা	-22.73	0.7273
দ্বীপ	উপদ্বীপ	ফল	-15.11	0.6499
গ্যাস	বাষ্প	বাড়ি	-85.44	0.5529

We observed from Table III two types of anomalies between the similarity score as obtained from word2vec and the priming result (or similarity perceived by human as recorded through psycholinguistics experiments): some word pairs which have high degree of priming but low word2vec similarity score and some word pairs having low degree of priming but high degree of word2vec similarity score. From the types of word pairs it is apparent that some word pairs, which we use together in our daily lives and colloquial use such as, **বিদ্যুৎ** and **গর্জন** . We also tend to foster a strong mental connection among them, that is reflected by the high DOP. On the other hand, as these word pairs are less likely to co-occur in a formal written corpus, they have low cosine similarity scores. Similarly, certain word pairs such as **মধুসূদন** and **রবীন্দ্রনাথ** or **গ্যাস** and **বাষ্প** have high word similarity score as they are likely to co-occur often due to their categorizations (first pair is names of poets and second pair is physics concepts), but not so much in day-to-day usage. Hence, our null hypothesis is proven to be wrong and we can infer that substantial gap still exists between how we *process* words and how computational approaches *think* we do.

IV. Conclusions and Discussion

In this paper we aim to study the representation and processing of semantically similar words in Bangla mental lexicon. Accordingly, we have conducted semantic priming experiment to determine the reaction time of subjects for prime-target and control-target pairs. We further computed the semantic similarity between the same word pairs using the Bangla word2Vec based word embedding model. We have compared the standard word embedding based semantic representation models correctly reflects the organization and processing of the mental lexicon. Analysis of reaction time indicates that corpus based semantic similarity measures does not reflects the true nature of mental representation and processing of words. The paper discusses various cases where it has been shown that highly similar words seldom shows any priming effect by the users

whereas word pairs having low corpus similarity may result in high degree of priming.

It is clear that the existing word embedding models are primarily based on the underline corpus on which they have been trained. Therefore, in order to understand the word representation strategies, ideally the corpus must bear a close resemblance with the human spoken form. However, such an ideal condition must always have to be approximated.

In particular, most of the natural language humans are exposed with belong to the spoken form. In order to use such data, it is required to do manual transcription of the spoken forms into their respective textual representations. This is not only time consuming but requires a huge man-power effort. On the other hand, the typical textual models that presently exists are based on written language that are available in the open web. Although, they are available in plenty, they are often less representative of the actual language input. In a recent work, Brysbaert et al. (2011) [13] showed that word frequency measures based on a corpus of 50 million words from subtitles predicted the lexical decision times of the English Lexicon Project [14] better than the Google frequencies based on a corpus of hundreds of billions words from books. Similar findings were reported for German [15]. In particular, word frequencies derived from non-fiction, academic texts perform worse [16].

Acknowledgment

The graduate students of Vidyasagar University and the twelve standard students of Gopali I.M.high school in India have actively helped us to performed these psycholinguistic experiments. So, we thank to all the participants.

References

[1] J. Grainger, P. Colé, and J. Segui, "Masked morphological priming in visual word recognition," *Journal of memory and language*, vol. 30, no. 3, pp. 370–384, 1991.

[2] E. Drews and P. Zwitserlood, "Morphological and orthographic similarity in visual word recognition." *Journal of Experimental Psychology: Human Perception and Performance*, vol. 21, no. 5, p. 1098, 1995.

[3] M. Taft, "Morphological decomposition and the reverse base frequency effect," *The Quarterly Journal of Experimental Psychology Section A*, vol. 57, no. 4, pp. 745–765, 2004.

[4] S. Dehaene, L. Naccache, G. Le Clec'H, E. Koechlin, M. Mueller, G. Dehaene-Lambertz, P.-F. van de Moortele, and D. Le Bihan, "Imaging unconscious semantic priming," *Nature*, vol. 395, no. 6702, p. 597, 1998.

[5] J. H. Neely, "Semantic priming and retrieval from lexical memory: Roles of inhibitionless spreading activation and limited-capacity attention." *Journal of experimental psychology: general*, vol. 106, no. 3, p. 226, 1977.

[6] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[7] K. Lund and C. Burgess, "Producing high-dimensional semantic spaces from lexical co-occurrence," *Behavior research methods, instruments, & computers*, vol. 28, no. 2, pp. 203–208, 1996.

[8] T. K. Landauer and S. T. Dumais, "A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge." *Psychological review*, vol. 104, no. 2, p. 211, 1997.

[9] K. I. Forster and C. Davis, "Repetition priming and frequency attenuation in lexical access." *Journal of experimental psychology: Learning, Memory, and Cognition*, vol. 10, no. 4, p. 680, 1984.

[10] K. Rastle, M. H. Davis, W. D. Marslen-Wilson, and L. K. Tyler, "Morphological and semantic effects in visual word recognition: A time-course study," *Language and cognitive processes*, vol. 15, no. 4-5, pp. 507–537, 2000.

[11] W. D. Marslen-Wilson, M. Bozic, and B. Randall, "Early decomposition in visual word recognition: Dissociating morphology, form, and meaning," *Language and Cognitive Processes*, vol. 23, no. 3, pp. 394–421, 2008.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[13] M. Brysbaert, E. Keuleers, and B. New, "Assessing the usefulness of google books' word frequencies for psycholinguistic research on word processing," *Frontiers in Psychology*, vol. 2, p. 27, 2011.

[14] D. A. Balota, M. J. Yap, K. A. Hutchison, M. J. Cortese, B. Kessler, B. Loftis, J. H. Neely, D. L. Nelson, G. B. Simpson, and R. Treiman, "The english lexicon project," *Behavior research methods*, vol. 39, no. 3, pp. 445–459, 2007.

[15] M. Brysbaert, M. Buchmeier, M. Conrad, A. M. Jacobs, J. Bölte, and A. Böhl, "The word frequency effect," *Experimental psychology*, 2011.

[16] M. Brysbaert, B. New, and E. Keuleers, "Adding part-of-speech information to the subtlex-us word frequencies," *Behavior research methods*, vol. 44, no. 4, pp. 991–997, 2012.

Development of a Bangla Sense Annotated Corpus for Word Sense Disambiguation

Monisha Biswas
Department of Computer Science & Engineering
Chittagong University of Engineering & Technology (CUET)
Chattagram-4349, Bangladesh
E-mail: monisha.cse10@gmail.com

Mohammed Moshiul Hoque
Department of Computer Science & Engineering
Chittagong University of Engineering & Technology (CUET)
Chattagram-4349, Bangladesh
E-mail: moshiulh@yahoo.com

***Abstract* – Sense annotated corpus can be treated as an essential resource for lexicon development, morphological processing and also for evaluating the performance of a word sense disambiguation (WSD) system. In this paper, a Bangla sense annotated corpus is generated from a raw collection of Bangla text, where only the sentences which contain at least one Bangla ambiguous word are retrieved from the raw corpus. All individual word forms of the sentences stored in our Bangla sense annotated corpus are tagged with their corresponding root word forms and POS types and the detected ambiguous words in the sentences are also tagged with their actual senses. The developed Bangla sense annotated corpus initially contains 5028 Bangla sentences with proper annotation and the overall performance of our Bangla sense annotated corpus creation system is 86.95%.**

Index Terms – Bangla language processing, Sense annotated corpus, Lexicon, Word sense disambiguation, Ambiguous word.

I. INTRODUCTION

A corpus can be considered as an empirical base for processing computational analysis of any natural language, which is a collection of texts in an electronic predefined form based on some criteria for linguistic research. A Bangla corpus can be used for several purposes such as for creating automatic dictionary, morphological processing, disambiguating an ambiguous word from a given context known as word sense disambiguation etc. Word sense disambiguation is the task of removing the lexical semantic ambiguity of an ambiguous word by assigning the appropriate sense to that ambiguous word in a given context. Basically WSD requires a set of meanings for each word to be disambiguated and a means to choose the correct one from that set [1]. Because of the scarcity of huge collection of Bangla annotated texts available in electronic format, we are lacking good Bangla WSD applications as well as other Bangla language processing applications.

In this work, we have tried to develop a Bangla sense annotated corpus which is a collection of validated words tagged with some features such as POS, stemming and detection of some popular Bangla ambiguous words tagged with their actual senses in different sentences for different feature words. A statistical analysis of the developed Bangla sense annotated corpus is also done in order to find out some statistical counting such as total number of occurrences of an ambiguous word, number of occurrences of each sense for an ambiguous word in the developed corpus etc., which measures are important for performing statistical based Bangla word sense disambiguation tasks.

This paper proposes a process of building an annotated Bangla corpus from a source of raw corpus, where corpus texts are collected from web resources. Our annotated corpus is essentially a list of huge number of sentences where all words are tagged with their root words, Parts-of-Speech (POS) tagging and detection of some popular ambiguous words in the sentences tagged with their actual meanings in a given context based on feature words. After that several types of statistical analysis is also done on our sense annotated corpus for promoting statistical approach based Bangla language processing tasks such as Bangla word sense disambiguation, machine translation and information retrieval etc.

II. RELATED WORK

In comparison to other languages like English, Chinese, Spanish etc. research advancement on Bangla corpus generation is still not so far. A significant number of research works have been done from the beginning of this millennium in [2, 3] for creation and analysis of Bangla corpus for achieving various linguistic properties of Bangla language.
An annotated Bangla news corpus is developed in [4], where the corpus contains 74,698 word forms and a lexicon of 14 thousand entries is provided where words are annotated with a combination of manual tags such as POS, Stemming and morphemes.

A Bangla text corpus BdNC01is created in [5], where nearly 12 million word tokens including literary corpus were collected and manually checked before preserving in final format as ASCII and Unicode text. It is a collection of large text in a uniform format followed by some statistical processing which gives some count and prior probabilities for probabilistic computation of language modeling. Because of unavailability of Bangla text corpus in the desired format of the researchers for their specific purpose, in most of the recent Bangla word sense disambiguation systems [6,7], researchers have used their own manually generated Bangla corpus for disambiguation of few Bangla words, where the corpus size is not mentioned. In [8], a better accuracy rate of word sense disambiguation system is achieved where a Bangla POS tagged corpus is used for the experiment which is collected from the Indian languages Corpora Initiative. So a properly

978-1-7281-5242-4/19

formatted large size corpus can improve the efficiency of any language processing tools and if it is generated for the purpose of any specific language processing applications, then corresponding researchers in their fields will be much benefitted. As Bangla word sense disambiguation is an emerging field and still requires good research activities, so in this work we are motivated to develop a Bangla sense annotated corpus which can be used efficiently for improving the accuracy of any BWSD tools as well as other Bangla language processing applications.

III. METHODOLOGY OF CREATING BANGLA SENSE ANNOTATED CORPUS

The process of creating Bangla sense annotated corpus involves several steps, which need to be followed thoroughly as illustrated in Fig. 1.

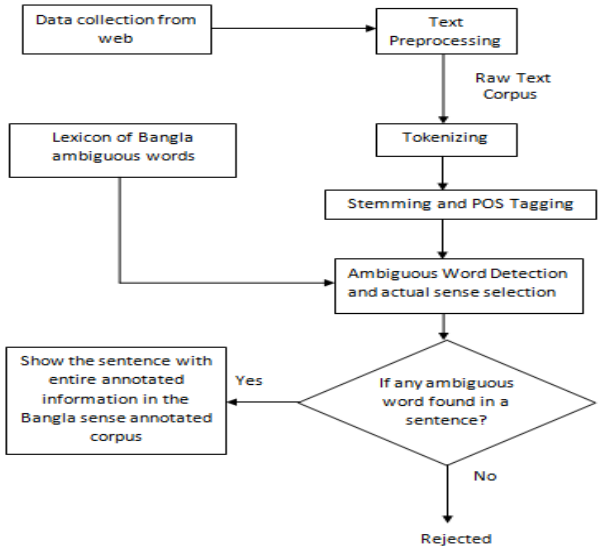


Fig. 1 Methodology of creating Bangla sense annotated corpus

A. Data Collection

Internet, the giant source of information, is used to collect Bengali text in electronic format for our raw text collection. As there is non-availability of good Bangla OCR applications in order to retrieve electronic formatted text from printed Bangla books and newspapers, so we have to proceed with online resources of web such as Bangla articles, blogs and news on several websites [9-16]. Firstly, few searches by some popular Bangla ambiguous words are performed on Google.com to collect top links of sources from search results. Secondly, HTTPS request is sent to every links using JavaScript "Request API" in order to fetch HTML contents. "JavaScript DOM API" is used to extract raw texts from HTML contents and afterwards, basic sanitization is done.

B. Preprocessing Text

The text collected from internet is non-normalized in nature which needs to be preprocessed for making the whole text normalized. So our collection of raw text is normalized by using the following steps as pre-processing:

i. First of all, three Bengali sentence termination symbols "।", "!" and "?" are considered for detecting individual Bangla sentences. For example, if we consider the Bengali text-"তাই যাওয়ার বেলায় তাকে জিজ্ঞেস করেছিলাম, তোমার শখ কী? উত্তরে সে বলেছিল, 'আংকেল মাঝেমধ্যে ছবি আঁকি'।"; then two sentences will be detected by our pre-processing process such as:

   1. তাই যাওয়ার বেলায় তাকে জিজ্ঞেস করেছিলাম, তোমার শখ কী?
   2. উত্তরে সে বলেছিল, 'আংকেল মাঝেমধ্যে ছবি আঁকি'।

ii. Different punctuation symbols such as Colon (:), Exclamation mark (!), Semi-colon (;), Apostrophe (')/Inverse Comma (')/Quotation Mark (''/""), Dash/Hyphen (-), Question mark (?), brackets ((),[]) are replaced by a single space except those which are preceded or followed by Bengali alphabet.

For example, উত্তরে সে বলেছিল, 'আংকেল মাঝেমধ্যে ছবি আঁকি'।

Here Comma (,) and Quotation Mark ('') have not been replaced by single space as they are preceded or followed by Bengali alphabet.

iii. Finally, Un-even numbers of newlines, blanks or spaces are removed from the raw text.

After applying all these pre-processing steps into our raw text, the final pre-processed text format in our raw corpus is like as below (refers Fig. 2):

ছেলে পরীক্ষায় পাশ করে পিতার মুখ রাখল, আজ সে তার পরিবারের মান রাখল। কথা বলে খুব ভাল লাগল ছেলেটিকে।তাই যাওয়ার বেলায় তাকে জিজ্ঞেস করেছিলাম, তোমার শখ কী? উত্তরে সে বলেছিল, 'আংকেল মাঝেমধ্যে ছবি আঁকি, আর অবসরে মাঝে মাঝে বই পড়তে পছন্দ করি।

Fig. 2. Pre-processesed text sample in our raw corpus

C. Tokenizing

Natural language toolkit has an important module called "tokenize", which can be either word tokenize or senntence tokenize. After pre-processing, word tokenization should be done in our work in order to split sentences into words. We split all the sentences of the raw corpus into individual words or tokens by the method "word-tokenize".For example, for a Bangla input sentence "ছেলে পরীক্ষাতে পাশ করে পিতার মুখ রাখল"; the output of tokenization will be like this ("ছেলে", "পরীক্ষাতে", "পাশ" , "করে", "পিতার" ,"মুখ","রাখল").

D. Stemming and POS Tagging

After tokenization stemming operation is done to convert a word into its constituent root part. Tokens are stored in an iterative list which is taken as input in this step. Here, first

suffix part of Bangla words are stripped off first, then the validity of this remaining word part as root word is checked using a Bangla dictionary. If a valid root word is not found yet, then the affix part of the remaining word form is stripped off. In this process, all the word forms of our raw corpus are converted into their root word forms and their corresponding suffixes and affixes are ignored for simplicity.

After that**,** each token is checked with an available machine readable online Bangla dictionary for its validity and its corresponding POS type is brought from the dictionary. Here, for Bangla language, we consider mainly six different parts-of-speech tags namely, noun, pronoun, verb, adverb, adjective and indeclinable (prepositions, conjunctions and interjections). Example of Bangla word list of several POS category with their corresponding Tag Name used in this system is given in Table I.

TABLE I: EXAMPLE LIST OF SEVERAL POS CATEGORIES

Tag Description	Tag Name	Examples
Noun	noun	মাথা, হাত, চোখ
Pronoun	pron	সে, আমি, তোমার, আমার, তাদের
Adjective	adj	অনেক, অধিক, ভালো
Verb	verb	করছেন, করল, আসা, ছিল
Adverb	adv	অবশেষে, তাহলে, সাধারণত
Indeclinable(Preposition, Conjunction & Interjection)	prep/conj/intj	হতে,থেকে,চেয়ে, এবং,কিন্তু

After assigning appropriate POS category to each word stem, the sample output of stemming and POS tagging process is given below in Fig. 3:

ছেলে/noun/ছেলে পরীক্ষাতে/noun/পরীক্ষা পাশ/adj/পাশ করে/verb/করা পিতার/noun/পিতা মুখ/noun/মুখ রাখল/verb/রাখা. আজ/adv/আজ সে/pron/সে তার/pron/ তার পরিবারের/noun/ পরিবার মুখ/noun/মুখ রাখল/verb/রাখা।

Fig. 3. Screenshot of sample text after stemming and POS tagging process

E. Lexicon of Bangla Ambiguous Words

In this work, a lexical database is manually developed to store the several senses of popular Bangla ambiguous words in presence of different feature words. The ambiguous word detection from the raw corpus will be done by using this lexicon of Bangla ambiguous words. It will also provide the actual sense of the detected ambiguous word based on its feature words to the ambiguity detector program. A screen shot of our lexicon of Bangla ambiguous words with their different meanings or senses in case of different feature words is given below in Table II. This storage of Bangla ambiguous words is developed fully manually through a long period of time by analysing different Bangla grammar books and websites [17-19].

TABLE II.SAMPLE OF DIFFERENT SENSES OF SOME BANGLA AMBIGUOUS WORDS IN CASE OF DIFFERENT FEATURE WORDS STORED IN THE LEXICON

Ambiguous Word	Corresponding Feature Word	Sense of Ambiguous Word
মাথা	আছে	বুদ্ধি
মাথা	ঠেকান	প্রনাম করা
মাথা	কাটা	লজ্জা পাওয়া
মাথা	গরম	চটে যাওয়া
মাথা	গ্রাম	প্রধান
মুখ	রাখা	সম্মান রাখা
মুখ	ভার	রাগ করা
মুখ	মুখ	তর্ক করা
মুখ	চাওয়া	নির্ভর করা
হাত	টান	চুরির অভ্যাস
হাত	নেই	প্রভাব
হাত	পাতা	সাহায্য চাওয়া
হাত	দেয়া	কাজ শুরু করা
হাত	করা	বশীভূত করা
হাত	তোলা	প্রহার করা

F. Ambiguous Word Detection and Actual Sense Selection

In our sense annotated corpus, only the sentences which have at least one ambiguous word will be stored from the raw corpus and all other sentences will be ignored. An ambiguity detector program is used in this work in order to detect ambiguous words from the raw corpus and it will also find out the actual sense of the detected ambiguous word with the help of our developed lexicon of Bangla ambiguous words.

The algorithm for ambiguous word detection from the raw corpus and also to find out the actual sense of the detected ambiguous words is given below:

Step 1: Sentences (S_1,S_2,....,S_n) from the raw corpus with POS tagging and stemming information (For example, as Fig. 3.) are set as input for the ambiguity detector program.

Step 2: For each sentence (S_i) of the raw corpus, repeat step 3 to 5.

Step 3: Individual word stem of the sentence (S_i) is checked one by one as an iterative list with the Bangla ambiguous word list stored in our lexicon in order to find out whether it is ambiguous or not.

For example if we consider the sentence, "ছেলে/noun/ছেলে পরীক্ষাতে/noun/পরীক্ষা পাশ/adj/পাশ করে/verb/করা পিতার/noun/পিতা মুখ/noun/মুখ রাখল/verb/রাখা."; Here from TABLE I, we can see that "মুখ" is stored as an ambiguous word in our lexicon of Bangla ambiguous word list. So our system will detect "মুখ" as an ambiguous word.

Step 4: If the ambiguity detector can able to detect any ambiguous word in the sentence, it will compare its immediate collocating root words in the both left and right side direction of the ambiguous word in the sentence with corresponding feature word list of that ambiguous word stored in the lexicon. If any match is found, it will provide the actual sense of the detected ambiguous word with the help of the knowledge given by the lexicon.

Else, that means if no match is found, the control will go back to step 3 and continue with the next sentence (S_{i+1}). For our above considered example sentence, when the ambiguity detector detects the ambiguous word "মুখ"; it will compare its

left and right collocating root words "পিতা" and "রাখা" with corresponding feature word list of the ambiguous word "মুখ" stored in the lexicon. From TABLE I we can see that, there are four feature words ("রাখা", "ভার", "মুখ", "চাওয়া") in the lexicon for the ambiguous word "মুখ". As the feature word "রাখা" is matched; the ambiguity detector will provide "সম্মান রাখা" as the actual sense of the ambiguous word "মুখ" for the above input sentence.

Step 5: When a sentence is found which contains an ambiguous word, this sentence is stored in our final Bangla sense annotated corpus along with POS type and root form of all word forms of the sentence and also the detected ambiguous word with its actual sense as following format:
ছেলে/noun/ছেলে পরীক্ষাতে/noun/পরীক্ষা পাশ/adj/পাশ করে/verb/করা পিতার/noun/পিতা [aw=মুখ, sense=সম্মান রাখা] মুখ/noun/মুখ রাখল/verb/রাখা।

Before adding this annotated information into our sense annotated corpus, the total occurrence value of the ambiguous word is increased by 1 for each time whenever it is detected. And also the numbers of occurrence of several senses of the ambiguous words are also recorded for further statistical analysis.

Step 6: Finally all the sentences from the raw corpus which contain Bangla ambiguous words will be stored in our Bangla sense annotated corpus with proper annotation.

IV. Experimental Results

A normal computer with Intel core i3 processor of 64 bit windows operating system along with 4GB RAM is used to implement the system using Python. There are tons of third-party packages available for Python to work with Natural Languages, likely NLTK (Natural Language Toolkit), TextBlob, Spacy, CoreNLP (Stanford CoreNLP) and so on. But all these packages are compatible only with English language. Almost zero third-party packages are available in the internet to work with Bangla. So our first problem was to build a Natural Language Toolkits for processing Bangla language to perform some basic operations like, text normalization, lemmatization, POS tagging etc. We have to build those functions at our own first. Then we have developed a system which takes input of Bangla raw corpus and provides a resulting Bangla sense annotated corpus as output. The following steps have been performed to evaluate the system:

A. *Sample Input Text after Text Pre-processing*

The text retrieved from the internet is normalized first by text preprocessing. A partial view of the normalized text is presented in Fig. 4.

জনাব আলম ভাল মানুষ। তিনি গ্রামের মাথা। সকলের ভাল চিন্তা করে।সহজে তাঁর মাথা গরম হয় না। তাঁর প্রথম ছেলে এবার পরীক্ষায় পাশ করে সমাজে পিতার মুখ রাখল। কিন্তু তাঁর দ্বিতীয় ছেলে কে একটু চোখে চোখে রাখতে হয়। কারন তার হাত টানের অভ্যাস আছে।

Fig. 4. Sample of normalized input data in the raw corpus

From this input data set, the system can detect total of 7 sentences by the Bangla sentence termination symbol "।". After that, tokenization, stemming and POS tagging processes are performed on it.

B. *Sample Text after Tokenization, Stemming and POS Tagging*

A sample text after tokenizing, stemming and POS tagging process is given in Fig. 5.

জনাব/noun/জনাব আলম/noun/আলম ভাল/adj/ভাল মানুষ/noun/মানুষ। তিনি/pron/তিনি গ্রামের/noun/গ্রাম মাথা/noun/মাথা সকলের/adj/সকল ভাল/adj/ভাল চিন্তা/noun/চিন্তা করে/verb/করা। সহজে/adv/সহজে তাঁর/pron/তাঁর মাথা/noun/মাথা গরম/adj/গরম হয়/verb/হয় না/adv/না। তাঁর/pron/তাঁর প্রথম/adj/প্রথম ছেলে/noun/ছেলে এবার/adv/এবার পরীক্ষায়/noun/পরীক্ষা পাশ/noun/পাশ করে/verb/করা সমাজে/noun/সমাজ পিতার/noun/পিতা মুখ/adv/মুখ রাখল/verb/রাখা। কিন্তু/conj/কিন্তু তাঁর/pron/তাঁর দ্বিতীয়/adj/দ্বিতীয় ছেলে/noun/ছেলে কে/pron/কে একটু/adv/একটু চোখে/noun/চোখ চোখে/noun/চোখ রাখতে/verb/রাখা হয়/verb/হয়। কারন/noun/কারন তার/pron/তার হাত/noun/হাত টানের/noun/টান অভ্যাস/noun/অভ্যাস আছে/verb/আছে।

Fig. 5. A sample text after tokenization, stemming and POS tagging

C. *Sample Output Data Set of Bangla Sense Annotated Corpus*

Our resultant Bangla sense annotated corpus will store only those sentences from the raw corpus which contain Bangla ambiguous words. From the above input data set, our system can detect 5 sentences out of 7 sentences which contain Bangla ambiguous words such as "মাথা", "মুখ", "চোখ", and "হাত" based on the knowledge from our lexicon of Bangla ambiguous words. That's why these 5 sentences are stored with proper annotation information into our resultant sense annotated corpus. A sample of output file generated by the system is given below in Fig. 6:

[1] তিনি/pron/তিনি গ্রামের/noun/গ্রাম [aw=মাথা, sense=প্রধান ব্যক্তি] মাথা/noun/মাথা

[2] সহজে/adv/সহজে তাঁর/pron/তাঁর [aw=মাথা, sense=চটে যাওয়া] মাথা/noun/মাথা গরম/adj/গরম হয়/verb/হয় না/adv/না

[3] তাঁর/pron/তাঁর প্রথম/adj/প্রথম ছেলে/noun/ছেলে এবার/adv/এবার পরীক্ষায়/noun/পরীক্ষা পাশ/noun/পাশ করে/verb/করা সমাজে/noun/সমাজ পিতার/noun/পিতা [aw=মুখ, sense=সম্মান রাখা] মুখ/adv/মুখ রাখল/verb/রাখা

[4] কিন্তু/conj/কিন্তু তাঁর/pron/তাঁর দ্বিতীয়/adj/দ্বিতীয় ছেলে/noun/ছেলে কে/pron/কে একটু/adv/একটু [aw=চোখ, sense=অঙ্গ বিশেষ] চোখে/noun/চোখ [aw=চোখ, sense=নজরে রাখা] চোখে/noun/চোখ রাখতে/verb/রাখা হয়/verb/হয়

[5] কারন/noun/কারন তার/pron/তার [aw=হাত, sense=চুরির অভ্যাস] হাত/noun/হাত টানের/noun/টান অভ্যাস/noun/অভ্যাস আছে/verb/আছে

Fig. 6. Sample output text in our Bangla sense annotated corpus

D. *Overall Results*

Initially our raw input corpus consists of 13241 Bangla sentences with a total of 358850 Bangla words from which our system can detect 5028 sentences which contain Bangla ambiguous words. That's why our system has retrieved these 5028 sentences from the raw corpus and stored them into the

Bangla sense annotated corpus with proper annotation information of each word form.

In order to evaluate our sense annotated corpus creation system, first we gave a Bangla text file of 200 Bangla sentences to some resource persons who have good background knowledge on Bangla language. According to their mutual discussions, 9 different Bangla ambiguous words have been found with their different senses for different feature words in the 43 sentences out of those 200 sentences. Then we have tested our system with the same data set and our system has generated a resultant Bangla sense annotated corpus where 7 different Bangla ambiguous words were detected and it can disambiguate these 7 detected ambiguous words in 36 sentences correctly. This procedure of experiment is repeated for two more times by changing the raw corpus file which contain 400 and 600 Bangla sentences respectively and the different values given by our system and experts are recorded. We have calculated the accuracy rate of our system on the basis of number of ambiguous words detected and the number of properly disambiguated sentences. Our accuracy rate denotes the ratio between the values given by our system and the values provided by the experts. TABLE III and TABLE IV illustrate the accuracy rate of our developed sense annotated corpus creation system for different number of sentences available in the raw corpus.

TABLE III. ACCURACY RATE ON THE BASIS OF NUMBER OF DETECTED AMBIGUOUS WORDS

Number of Sentences in the Raw Corpus	Number of Ambiguous Words to be Detected	Number of Detected Ambiguous Words by the System	Accuracy Rate	Overall Accuracy
200	9	7	77.78%	87.25%
400	12	11	91.67%	
600	13	12	92.31%	

TABLE IV. ACCURACY RATE ON THE BASIS OF NUMBER OF CORRECTLY DISAMBIGUATED SENTENCES

Number of Sentences in the Raw Corpus	Number of Sentences Found to Disambiguate	Number of Sentences Disambiguated Correctly by Our System	Accuracy Rate	Overall Accuracy
200	43	36	83.72%	84.95%
400	61	52	85.25%	
600	78	67	85.89%	

The graphical representation of the accuracy rate vs. number of input sentences in the raw corpus is given below in Fig.7.

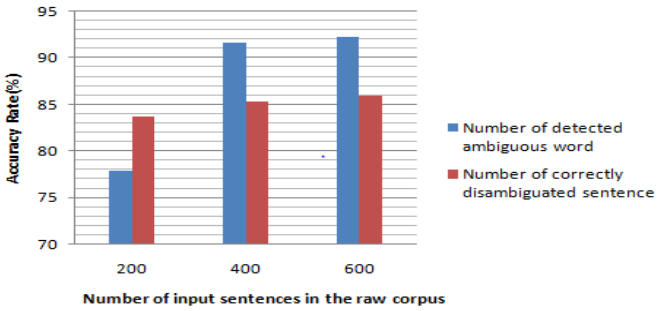


Fig. 7. Accuracy Rate vs. number of input sentences in the raw corpus

The proposed system is developed for automatic generation of Bangla sense annotated corpus from any raw collection of Bangla texts with an overall accuracy of 86.95%.

E. Statistical Analysis

We also focus on statistical analysis of the detected Bangla ambiguous words from the raw corpus in order to help Bangla researchers in developing an efficient word sense disambiguation tool using statistical approach. So we have developed our sense annotated corpus in such an efficient way that it provides some automated query based statistical counting for all the detected ambiguous words of the corpus such as "মাথা","গরম", "চোখ", "হাত", "মন","মুখ", "কড়া", "তোলা", "উঠা", "লাগা", "বুক", "পা" etc. For example, for the ambiguous word "মাথা", following types of statistical counting are given by system as shown in TABLE V.

TABLE V. STATISTICAL COUNTING FROM OUR BANGLA SENSE ANNOTATED CORPUS FOR AMBIGUOUS WORD "মাথা"

Ambiguous Word	Total No. of Occurrence of Ambiguous Word Found	Feature Word	Sense of Ambiguous Word	No. of Occurrence of Sense Found
মাথা	144	গাঁয়ের	প্রধান ব্যক্তি	47
		ব্যথা	শারীরিক অসুস্থতা	16
		গ্রামের	প্রধান ব্যক্তি	47
		এলাকার	প্রধান ব্যক্তি	47
		শহরের	প্রধান ব্যক্তি	47
		আছে	বুদ্ধি	2
		ঠেকান	প্রণাম করা	69
		আসা	বোধগম্য হওয়া	4
		কাটা	কাটা	6
		খাওয়া	নষ্ট করা	10
		পাতা	সম্মত হওয়া	2
		দেওয়া	সাহায্য দেওয়া	10
		চোখের	অন্ধ হওয়া	10
		গরম	চটে যাওয়া	15

VI. Conclusion

Sense annotated corpus can be an essential resource for the training phase of a supervised word sense disambiguation system. Indeed, a large size of Bangla sense annotated corpora may be a vital resource for many emerging fields of Bangla language processing. The main purpose of this work is to facilitate the construction and evaluation of Bangla WSD systems by providing a Bangla sense annotated corpus in a well defined format from collection of huge Bangla raw texts. Through our system, any size of Bangla raw text file can be taken as input to develop a corresponding Bangla sense annotated corpus in a well defined format as output. Several statistical calculations of the detected ambiguous words in the sense annotated corpus are also given by our system so that researchers will be greatly benefited in future to develop a corpus based Bangla WSD system using statistical approach. As the performance of any knowledge based system mainly depends on the size of its resource, so in near future we will try to collect more collection of Bengali text as raw corpus in order to increase the size of our Bangla sense annotated corpus, so that it would be a vital resource for Bangla language related research activities.

References

[1] A. Hoque and M. M. Hoque, "Bangla word sense disambiguation using dictionary based approach ", in *Proc. Int. Conf. on Advanced Information & Communication Technology*, 2016.

[2] N. S. Dash and B. B. Chaudhuri, "A corpus based study of the Bangla language", Indian Journal of Linguistics, vol. 20, pp. 19-40, 2001.

[3] N. S. Dash and B. B. Chaudhuri, "Corpus generation and text processing", International Journal of Dravidian Linguistics, vol. 31(1), pp. 25-44, 2002.

[4] T. H. Chaudhury, A. Matin, M.S. Hossain, A. Uzzaman and M. Masum, "Annotated Bangla news corpus and lexicon development with POS tagging and stemming", The Global Journal of Researches in Engineering, vol. 17, no. 1-J, 2017.

[5] M. F. Khan, A. Ferdousi, M. A. Sobhan, "Creation and analysis of a new Bangla text corpus BDNC01", International Journal for Research in Applied Science & Engineering Technology (IJRASET), vol. 5, November 2017.

[6] S. Nazah, M. M. Hoque, R. Hossain, "Word sense disambiguation of Bangla sentences using statistical approach", in *Proc. of 3rd International Conference on Electrical Information and Communication Technology,* December 2017.

[7] A. R. Pal, D. Saha, S. Naskar, "Word sense disambiguation in Bengali: A knowledge based approach using Bengali WordNet ", in *Proc. of IEEE 2nd International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, February 2017.

[8] A. R. Pal, D. Saha, N. S. Dash, A. Pal, "Word Sense Disambiguation in Bangla Language Using Supervised Methodology with Necessary Modifications", Journal of The Institution of Engineers (India), vol. 99(5), pp. 519-526, 2018.

[9] https://bn.wikipedia.org

[10] https://www.banglapedia.org

[11] https://www.bbc.com/bengali

[12] https://www.prothomalo.com

[13] https://www.jugantor.com

[14] https://www.ittefq.com.bd

[15] https://samakal.com

[16] https://www.dailynayadiganta.com

[17] Humayun Azad, Bakyatattya, The University of Dhaka, Dhaka, 1995.

[18] https://eshikhon.com

[19] http://www.grammarbd.com

Automatic Detection of Satire in Bangla Documents: A CNN Approach Based on Hybrid Feature Extraction Model

Arnab Sen Sharma
Computer Science and Engineering
Shahjalal University of
Science & Technology
Sylhet-3114, Bangladesh
Email: arnab.api@gmail.com

Maruf Ahmed Mridul
Computer Science and Engineering
Shahjalal University of
Science & Technology
Sylhet-3114, Bangladesh
Email: mridul-cse@sust.edu

Md Saiful Islam
Computer Science and Engineering
Shahjalal University of
Science & Technology
Sylhet-3114, Bangladesh
Email: saiful-cse@sust.edu

***Abstract*—Wide spread of satirical news in online communities is an ongoing trend. The nature of satires are so inherently ambiguous that sometimes it's too hard even for humans to understand whether it's actually satire or not. So, research interest has grown in this field. The purpose of this research is to detect Bangla satirical news spread in online news portals as well as social media. In this paper we propose a hybrid technique for extracting feature from text documents combining *Word2Vec* and *TF-IDF*. Using our proposed feature extraction technique, with standard CNN architecture we could detect whether a Bangla text document is satire or not with an accuracy of more than 96%.**

***Index Terms*—satire detection, natural language processing, TF-IDF, fact-checking, CNN, Word2Vec.**

I. Introduction

Satires can be considered as a literary form which involves a delicate balance between criticism and humor. Through satire or sarcasm, messages are conveyed in an artistic form that sometimes creates a deviated implicit meaning. The goal of satire is not always to tell the truth. Sometimes humans are not effective enough to distinguish between satires and actual news because often the satires are so ambiguous that it is easy to get deceived.

The spread of satirical news is not a new concept. But in the recent years it has become a real threat that can not be ignored anymore. Easy access of Internet and hyperactivity of users in various social media platforms has given rise to the extensive spread of satirical news. Internet has largely replaced traditional news media. Many people, especially a huge portion of youth depend on Internet and social media as the primary source for news consumption because of their easy access, low cost and 24/7 availability. They simply believe in what they read in internet and spread the news what they assumed to be true. So, most of the times, satires are not spread with an intention to deceive. But sometimes some people for their personal benefits take advantage and promote the spread of satires as actual news.

As a matter of fact, there are some web based applications such as Snopes.com, FactCheck.org, PolitiFact etc which act as fact-checkers. But, these services use human staffs to manually check facts. Though these services provide accurate information most of the time, these are not efficient enough since they are not automated. We propose an automated system based on Convolutional Neural Network and Natural Language Processing to address the problem.

There are some related existing works. The *Literature Review* section will discuss about these. Also, we'll define some terms and techniques that we used in this work.

A. Literature Review

De Sarkar, et al. proposed a hierarchical deep neural network approach to detect satirical fake news which is capable of capturing satire both at the sentence level and at the document level [1].

Burfoot, et al. used SVM and bag-of-words to detect satires [2]. They used binary feature weights and bi-normal separation feature scaling for feature weighting. They got a best overall F-score of 79.8%[2].

Rubin, et al. classifies news as *Satires, Fabrications and Hoaxing* as the parts of fake news [3].

Reyes, et al. used figurative language processing for humour and irony detection [4].

Ahmad and Tanvir used tockenized, stopword free and stemmed data to classify satire and irony using SVM and got an accuracy of 83.41%[5].

el Pilar Salas-Zrate and Mara used some psycholinguistic approaches for satire detection in twitter and got F-score of 85.5% for mexican data and 84.0% for spanish data [6].

Tacchini and Eugenio check facts using the information of the users who liked a news[7]. Applying logistic regression on the information of likers, around 99% accuracy is achieved for their dataset.

Some approaches simply used a naive bayes classifier to classify a news. After a little bit of preprocessing on 1-grams from the news context, words are fed to a naive bayes classifier. Granik, et al. proposed to do a stemming to increase accuracy[8]. The accuracy got using this approach without

978-1-7281-5242-4/19

preprocessing is nearly 70%[8] and Pratiwi, et al. got accuracy of 78.6% with preprocessing in Indonesian Language[9].

Ruchansky, et al. proposed a Capture Score and Integrate model[10]. Sense making words from the body text of the news of *twitter*[11] and *weibo*[12] are taken and fed to a RNN and the reviews of the news are taken as feature. The accuracy for this model is 89.2% for *twitter* data and 95.3% for *weibo* data[10].

Conroy,et al. proposed two different approaches for detecting fake news [13].

One is linguistic approach which includes deep syntax analysis and semantic analysis. Deep syntax analysis is implemented based on Probability Context Free Grammars(PCFG). It can predict falsehood approximately 91% accurately.

Another one is network approach that is based on fact checking using the knowledge networks formed by interconnecting the linked data. This approach gives an accuracy in the range of 61% to 95% for different subject areas.

B. Definitions

1) Word Embedding: Word embedding simply refers to vector representation of words. Normally machine learning models are not capable of processing string or text as input. These models expect vectors or values as input. So, transformation of a word to a vector is a crucial part. There are several techniques to convert word to vectors. These techniques can be categorized as two types 1. Frequency based (TF-IDF, CountVectorizer, HashVectorizer) 2. Prediction based (Word2Vec)

2) TF-IDF: In this work we focused on TF-IDF vectorizer amongst TF-IDF, CountVectorizer, HashVectorizer etc. as the frequency based word embedder. TF stands for *Term Frequency* and IDF stands for *Inverse Document Frequency*. It is used in text mining as a weighting factor for features. The equation representing the TF-IDF weight of a term t for a particular document d (given the dataset D and the number of documents in the dataset N) is

$$tf - idf(t,d) = tf(t,d) \times idf(t,D)$$
$$Here,$$
$$tf(t,d) = \text{Frequency of } t \text{ in } d$$
$$idf(t,D) = log(\frac{N}{|\{d \epsilon D : t \epsilon d\}|}) \text{ [14]}$$

TF is upweighted by the number of times a term occurs in an article. And IDF is downweighted by the number of times a term occurs in the whole dataset/corpus. So TF-IDF assigns less significant values to words that generally occurs in most documents such as *is, are, be, to, on ... etc*.

3) Word2Vec: Though heavily used in the field of NLP, frequency based word embedders fail to capture the semantic value of a word or document. Word2Vec is the process of transforming words to vectors preserving some of their syntactical and semantic correlations. Word2Vec tries to determine the meaning of a word and understand its correlation with other words by looking at its context. For example, lets take two sentences "Range Rover is great car" and "Range Rover is a wonderful vehicle", then Word2vec will map similarities between the words *great* and *wonderful* and the words *car* and *vehicle*. Word2Vec uses cosine distance over euclidean distance to measure the similarity or distance between two words. Let's take some pair of singular-plural words like *cat* and *cats*, *dog* and *dogs*. Here the singular-plural relation between the words cat and cats is represented by the cosine difference between the two words V_{cat} and V_{cats} (1) is given by the equation below

$$cosine(V_{cat}, V_{cats}) = \frac{V_{cat} \times V_{cats}}{||V_{cat}|| \times ||V_{cats}||}$$

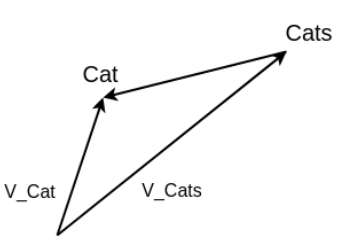


Fig. 1: Word2Vec (a)

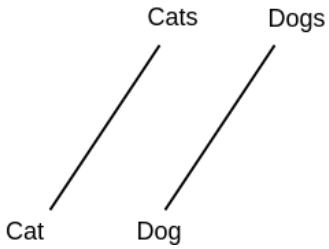


Fig. 2: Word2Vec (b)

And now the word pair dog and dogs have the same singular-plural relationship between them like cat and cats (2). So according to Word2Vec

$$V_{dog} - V_{dogs} = V_{cat} - V_{cats}$$
$$\Rightarrow V_{dogs} = V_{cats} - V_{cat} + V_{dog}$$

So, if we know the particular relationship between two words and know one of the words, Word2vec can predict the other word.

4) CNN in NLP: Convolutional Neural Network(CNN) is a deep learning algorithm that takes a multidimentional vector or an image as input. CNNs captures the significant aspects of the input through the application of appropriate *filters* and perform classification tasks. With enough training CNNs are able to learn which filters are appropriate for different contexts. Different filters/kernels are slided on the beginning layers named after convolutional layer and extract different features and feed-forward the values to next layer of the architecture. Besides learning high level features of an image these filters also reduce the size of convolved feature space and thus reduce the computational power required for processing the data. The convolved features extracted by the convolutional layers are then fed to a normal neural network architecture possibly with a number of hidden layers. This neural network learns the convolved feature vector and does the actual classification task.

Recently CNNs are used heavily in NLP. CNN expects the input to be a multidimensional image but we have a one dimensional vector from a word after word embedding. So, instead of single words we feed whole sentences or documents into CNN . From a 20 word document or sentence where

each word is embedded to a 200 dimension vector, we can get a 20x200 matrix. This two dimensional vector can act as an image and can be fed as an input to CNN. From various experiments and researches it was found that CNN performs quite well in generalizing the relationships between words in a document and thus capturing itś semantic meaning. CNN performs better than the simple bag of words and prone to less inaccurate assumptions.

II. Proposed method

First of all, we built our own Word2Vec model modifying the traditional Word2Vec model. Then an image is created from a preprocessed document combining Word2Vec and TF-IDF vectorizers. Finally, that image is used as input to a CNN architecture. The detailed procedure is given below.

A. Building our own Word2Vec model

There are some great Word2Vec models for English language, but as per our knowledge there is no good performing Word2Vec model for Bangla. So, we had to create our own Word2Vec model. To do so, we relied on *gensim* library of python. We needed a lot of textual data to train the model. We used the *scrapy* library of python to build crawlers and crawled Bangla textual data from Wikipedia and online news portals. We collected 380832 articles in total and used these to train our model. Our Word2Vec model converts Bangla words to a vector of size 10. To check the performance of our model, we checked the 5 most similar words of a Bangla word. Here are the results.

main word	similar words
খবর	'দুসংবাদ' - 0.501 'খবরাখবর' - 0.475 'প্রতিবেদন' - 0.456 'গুজব' - 0.446 'বরা' - 0.443

Fig. 3: Testing Word2Vec model with a word

B. Dataset

We used the *scrapy* library of Pyhon to build crawlers to crawl bangla textual data from different websites. For authentic news data we crawled news articles from two bangla news portals *Prothom Alo*[15] and *Ittefaq*[16]. For satire data we crawled articles from a renowned satire news portal *Motikontho*[17]. We crawled a total of 1480 articles from *Motikontho*. To balance our dataset we randomly selected 1480 articles from the REAL news articles we collected from *Prothom Alo* and *Ittefaq*. The formation of the dataset is very simple. A single data is only a document and a label (satire or not).

C. Data Preprocessing

The collected dataset might be mixed with some noisy and unnecessary data. So, we had to get rid of them through a bit of preprocessing. Our preprocessing consisted of the following steps.

1) Ignoring stopwords and punctuations: We ignored the stopwords from the news documents. Because, these words appear in almost every article and do not provide any significant information. Some example of stopwords -

ও, এবং, আর, এ, একটি, তাই etc

We collected the list of Bangla stopwords from genediazjr[18].

2) Stemming: We used a stemmer developed by Rafi Kamal[19] which we found better than other existing Bangla stemmers.

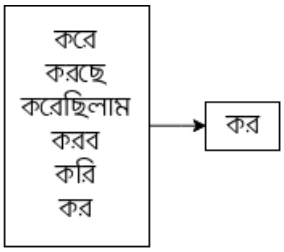


Fig. 4: Stemming

D. Document to Image

For converting a document to a vector we selected TF-IDF vectorizer of size 1000. It means 1000 most significant words in our corpus is selected and their respective TF-IDF values of a document is used to create a vector of size 1000 that represents the document. But these TF-IDF values do not hold any semantic meanings. So each of the words were converted to a vector of size 10 using our Word2Vec model. These vectors of size 10 were multiplied by their respective TF-IDF values for a document. So, for each document we had a 2D vector of size 1000×10. Also, convolutional layers expect pixel values of an image. Pixel values are never negative. So, CNNs cannot take a vector which contains some negative values. But word2vec embedded vectors can contain negative values. CNN input layers actually take a 3 dimensional vector. First two dimensions represents the 2D image and the third dimension is the number of filters. For coloured images the number of filters is 3(Red, Green, Blue) and each pixel value of image is formed with 3 values in RGB system. Our strategy to handle negative numbers was to separate the positive and negative values in two dimensions just like the picture below.

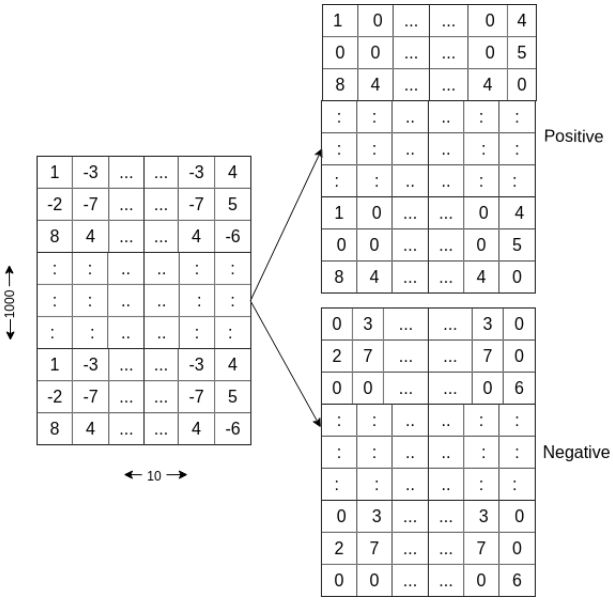


Fig. 5: Transformation of Feature Vectors

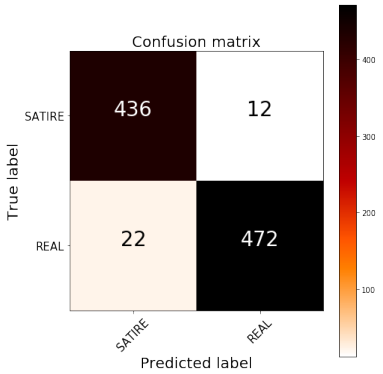
So, for each document we had a 3D vector of size $1000 \times 10 \times 2$.

E. Structure of the model

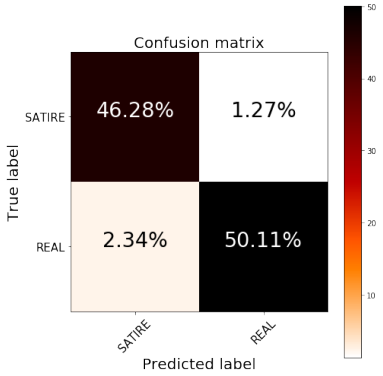
- Input layer: Input layer contains a Convolutional2D layer with 256 filters that takes vectors of shape $1000 \times 10 \times 2$ as input.
- Another Convolutional2D layer with 128 filters with *ReLU* activation.
- Pooling layer of size 2×2.
- Dropout layer with value 0.25 to avoid overfitting.
- A Dense layer with 512 neurons with *ReLU* activation.
- Dropout layer with value 0.5.
- Output layer: One neuron with *sigmoid* activation.

III. Results and Analysis

The dataset was random split to two parts. 70% of the data (2018 documents) were used as training dataset. The rest 30% (942 documents) were used to test the performance of our model. This model gave us an accuracy rate of 96.4% on the test dataset. Since the dataset was balanced the F1 score was same as the accuracy value. The confusion matrix is given below.



(a) Total Count Representation



(b) Percentile Representation

Fig. 8: Confusion Matrices showing the results

There are some scopes to improve our Word2Vec model and a perfect stemmer for Bangla language can boost the overall performance of our proposed model. The model was compiled with 2GB graphics card of NVIDIA GeForce 940M. So, we could not use a TF-IDF vector and a Word2Vec vector of larger size. If we had access to more resources we could use bigger feature vectors and the accuracy might have improved some.

Actually, in terms of accuracy, humans are much more effective than a machine for this task. Maybe for our dataset, human will be able to detect satires 100% accurately. But, though the accuracy falls a bit, we think it's much better to use an automated approach which saves a lot of time.

IV. Conclusion

Satire detection for Bangla news is completely new. As per our knowledge, no such work has been done in this sector for Bangla language. We found that our hybrid feature extraction technique combined with a CNN model performs great in language processing for pattern finding. Since satire is a type of fakeness, satire detection can be an important prerequisite of fake news detection. So, this work might be helpful to take better decisions on fake news detection and other such works. Also our hybrid feature extraction technique can be used in other works similar nature.

REFERENCES

[1] De Sarkar, Sohan, Fan Yang, and Arjun Mukherjee. "Attending sentences to detect satirical fake news." *In Proceedings of the 27th International Conference on Computational Linguistics (pp. 3371-3380).*

[2] Burfoot, Clint, and Timothy Baldwin. "Automatic satire detection: Are you having a laugh?." *Proceedings of the ACL-IJCNLP 2009 conference short papers. 2009.*

[3] Rubin, Victoria L., Yimin Chen, and Niall J. Conroy. "Deception detection for news: three types of fakes." *Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community. American Society for Information Science, 2015.*

[4] Reyes, Antonio, Paolo Rosso, and Davide Buscaldi. "From humor recognition to irony detection: The figurative language of social media." *Data & Knowledge Engineering 74 (2012): 1-12.*

[5] Ahmad, Tanvir, et al. "Satire detection from web documents using machine learning methods." *2014 International Conference on Soft Computing and Machine Intelligence. IEEE, 2014.*

[6] del Pilar Salas-Zrate, Mara, et al. "Automatic detection of satire in Twitter: A psycholinguistic-based approach." *Knowledge-Based Systems 128 (2017): 20-33.*

[7] Tacchini, Eugenio, et al. "Some like it hoax: Automated fake news detection in social networks." *arXiv preprint arXiv:1704.07506 (2017).*

[8] Granik, Mykhailo, and Volodymyr Mesyura. "Fake news detection using naive Bayes classifier." *2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON). IEEE, 2017.*

[9] Pratiwi, Inggrid Yanuar Risca, Rosa Andrie Asmara, and Faisal Rahutomo. "Study of hoax news detection using nave bayes classifier in Indonesian language." *2017 11th International Conference on Information & Communication Technology and System (ICTS). IEEE, 2017.*

[10] Ruchansky, Natali, Sungyong Seo, and Yan Liu. "Csi: A hybrid deep model for fake news detection." *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. ACM, 2017.*

[11] https://twitter.com/
Last accessed: 1:50 AM , 8/12/2018

[12] https://www.weibo.com
Last accessed: 1:50 AM , 8/12/2018

[13] Conroy, Niall J., Victoria L. Rubin, and Yimin Chen. "Automatic deception detection: Methods for finding fake news." *Proceedings of the Association for Information Science and Technology 52.1 (2015): 1-4.*

[14] https://en.wikipedia.org/wiki/Tf-idf
Last accessed: 8:55 PM , 03/08/2019

[15] https://www.prothomalo.com/
Last accessed: 8:59 PM , 03/08/2019

[16] https://www.ittefaq.com.bd/
Last accessed: 8:59 PM , 03/08/2019

[17] https://motikontho.wordpress.com/
Last accessed: 9:01 PM , 03/08/2019

[18] https://github.com/stopwords-iso/stopwords-bn/blob/master/stopwords-bn.txt
Last accessed: 7:38 PM , 03/08/2019

[19] https://github.com/rafi-kamal/bangla-stemmer
Last accessed: 7:51 PM , 03/08/2019