Bangla Sentence Correction Using Deep Neural Network Based Sequence to Sequence Learning

Sadidul Islam
Mst. Farhana Sarkar
Towhid Hussain
Md. Mehedi Hasan
Department of Computer Science and Engineering
United International University

A thesis submitted for the degree of
BSc in Computer Science & Engineering

September 2018

Abstract

Development in deep neural networks in particular to natural language processing has motivated researchers to apply these techniques in solving challenging problems like machine translation, automatic grammar checking, etc. In this paper, we address the problem of Bangla sentence correction and auto completion using decoder-encoder based sequence-to-sequence recurrent neural network with long short term memory cells. For this purpose, we have constructed a standard benchmark dataset incorporating mis-arrangement of words, missing words and sentence completion tasks. Based on the dataset we have trained our model and achieved 79% accuracy on the test dataset. We have made all our methods and datasets available for future use of the other researchers from: `https://github.com/mrscp/bangla-sentence-correction`. An online tool have also been developed based on our methods and readily available to use from: `http://brl.uiu.ac.bd/s2s`.

Acknowledgements

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

Contents

List of Figures

Chapter 1

Introduction

With upgrading of technology, our textual communication through the technology is increasing day by day. There has been a huge growth among the number of users in the online community and social media that uses Bangla as medium of communication. Error detection and correction in text based Bangla sentence correction using machine learning techniques is now more relevant due to availability of text online more than any time in the past. Among the traditional methods used for this purpose are morphological analysis [1, 2], Hidden Markov Models [3], active learning [4], natural language generation based approach [5], n-gram based methods [6] etc.

In contrast to the traditional methods, very recently recurrent neural networks (RNN) [7] and long short term memory (LSTM) networks have gained much success in natural language processing tasks. The advent of graphical processor unit (GPU) based computations have even accelerated the research in this field. Although they have been used in the context of many other languages [8, 9] providing innovative solutions to several problems like text correction [10], sentiment analysis [11], personality detection [12], etc., the development is not much found in the context of Bangla language. The application of deep learning in the context of Bangla Computing has been so far limited to speech recognition [13] and hand written digit recognition [14].

Even though there had been a number of work in the literature of natural language processing and computing in Bangla, still there are a number of challenges in the field that needs to be addressed [15–17]. In the literature for Bangla Natural Language Processing and Bangla Computing in general, we have observed a number of difficulties that are added to the challenges in the research and development. Firstly, there is lack

of publicly available standard datasets. Most of the researchers have worked on very small sized datasets and most of them are not available to future use. There is a need of publicly available datasets prepared with a standard methodology. Secondly, the methods proposed in the literature are not available for use or for research purpose. Most of the authors do not provide their programs or software to make the research usable, extendible and reproducible for the other researchers. Lastly, methods are not implemented or available for use of the general users outside the research community.

In this paper, we have addressed all the three challenges faced by any researcher or user of Bangla Computing tools specially in the field of Bangla Natural Language Processing. We address the problem of Bangla Sentence error correction and auto-completion. Firstly, we have constructed a large dataset using a controlled randomized procedure. Our benchmark dataset contains several types of errors like missing words, misplaced words and wrongly arranged words. Based on the dataset that we have used, we have implemented a sequence to sequence learning model that takes a wrong sentence as input and predicts a correct sentence for it. Our proposed method uses a encoder-decoder architecture sequence-to-sequence of deep learning networks. Here, the decoder is a bidirectional dynamic recurrent neural network with long short term memory (LSTM) cell. After training with the constructed dataset, our model was able to achieve 79% accuracy on the test set. We have also implemented a web based tool based on our model which is capable of correcting given input sentences. The tool for automated sentence correction is available to use from `http://brl.uiu.ac.bd/s2s`. We have also made our methods and datasets used in this paper available for other users for the sake of scientific purpose from `https://github.com/mrscp/bangla-sentence-correction`. The main contributions made in this paper are enumerated as following:

1. We have constructed a benchmark dataset for Bangla sentence correction from real sentences by introducing noise of several types in a controlled way.

2. We have proposed a deep sequence to sequence model for sentence correction using recrurrent network based encoder-decoder model.

3. We have trained our model using GPU based system to reduce the training time on the data.

4. We have developed an online tool for sentence correction and made all our materials and methods available online for future use by other researchers.

To the best of our knowledge, this is a first kind of research work on Bangla sentence correction using deep neural networks. We believe our data, methodology and the developed tool for Bangla sentence correction will serve the researchers and the general users in the future.

Rest of the paper is organized as follows: Chapter 2 briefly describes the related work in the literature; Chapter 3 presents our methods; Chapter 4 presents experimental results and Chapter 5 concludes the paper with a summary and a direction for future work.

Chapter 2

Related Work

There had been a number of research work on Bangla grammar checking [3, 5], parts of speech tagging [18] and in the field of general natural language processing. Traditional methods include methods like morphological analysis [1, 2], Hidden Markov Models [3], active learning [4], natural language generation based approach [5], n-gram based methods [6] etc. One of the earlier work done on Bangla computing was by Sengupta et al. in [2] where the authors used morphemes and finite state automata for automatic spelling error correction in Bangla sentences. Sajib Dasgupta and Vincent Ng [1] used unsupervised methods of morphological learning and achieved superior results on supervised parsers for Bangla language. The dataset they used had 4110 words.

A rule based filter with Hidden Markoc Models based parts-of-speech tagger was used in [3] to construct valid sentences from synthetically created erroneous sentences. They have used substitution, insertion, deletion and transposition of words in a valid sentence and used a dataset with nearly half million sentences collected from websites. In a subsequent work [4], the same authors proposed a complexity measurement metrics for grammar correction in Bangla sentences. They have made their grammar checker and complexity tester methods online. However, these methods are no more available to test.

Alam et al. [6] used n-gram based methods for grammar checking in a sentence. They have used a dataset of 866 sentences and achieved 63% accuracy using n-gram analysis of words and part-of-speech tagging for grammar checking. Kundu et al. [5] used a natural language generation approach to detect and correct errors in Bangla sentences. Hasan et al. [18] presented a review and comparison among different parts-

of-speech tagging methods. Parts-of-speech tagging is one of the pre-steps in grammar checking models used by the earlier group of researchers. Another prominent direction in the Bangla natural language processing is the sentiment analysis. Das et al. [19] proposed a lexicon of Bangla phrases called 'SentiWordNet' following the similar approaches done for English. Sarker et al. [20] proposed a sentiment analysis system for Bangla tweets using a Naive Bayesian model. Their proposed model also works for other majod Indian language, Hindi.

With the advent of Graphical Processing Unit (GPU) based computing techniques that enabled parallel computation and several new techniques for training and representation of neural networks deep learning has become very popular and effective in recent years and has been widely use to solve various problems in natural language processing where traditional machine learning methods have been used earlier [10]. One of the major advantage in deep learning is that the feature engineering is now part of the classifier and its representation and thus the layers of the neural networks are designed in a way to generate inherent properties of the grammar and structure of the languages. The techniques that were earlier used exclusively in other application areas like image processing, speech recognition, etc are now being tested with success in natural language processing [8–12].

Ghosh et al. [10] proposed a sequence to sequence architecture for text correction. They have used character based convolutional neural network gated recurrent units as encoder and word based recurrent units for decoders and used them for text correction and auto-completion in keyboard decoding. Poria et al. [8] used deep convolutional networks for multi-modal sentiment analysis of short video clips describe in one sentence. Santos et al. [11] used feed forward architecture with convolutional network to show the effectiveness in sentiment analysis on short text messages. Document based features were used in [12] to feed them into a convolutional neural network where the extracted features were used in a hierarchical manner to detect five personality traits. Joshi et al. [9] used long short term memory based architecture for sentiment analysis on mixed Hindi-English texts.

The use of Deep Neural Networks are not much found in the Bangla language processing. However, a few authors have used them for other fields in the Bangla computing domain like handwritten character recognition [14], speech recognition [13], etc. A two step deep belief network was used in [14] for recognition of handwritten Bangla

characters. In the first stage, deep belief networks were employed for unsupervised feature learning followed by a supervised learning stage where network parameters are learned by fine tunning.

Chapter 3

Proposed Method

In this chapter, we present the methodology of our paper. A system diagram of our methodology is given in Fig. 3.1. The first step starts with Bangla content collection from various sources followed by a pre-processing step, where the sentences are collected and converted to be applicable for a Word2Vector representation. Each sentence goes under perturbation for noise generation in the input valid sentences. After that, a Word2Vector representation is made for each of the sentences which are feed to a sequence to sequence model [21]. The trained model is then used to predict correct sentences from a given input wrong sentence using inference model. Rest of this section describes each step followed in the methodology in detail.

3.1 Bangla Content Collection

We collected Bangla stories, posts, news from various website through the internet. Currently, our dataset contains about only 250K of sentences. These sentences were collected from 3000 stories, posts and news collected from the web. Fig.3.2 shows a histogram of length of sentences collected in this step. We could see that maximum length of the sentences were 20. Two websites were used to collect the content:

1. Protom Alo: The most popular Bangla news paper (`https://www.prothomalo.com/`).

2. Somewherein Blog: The most popular Bangla blog (`http://www.somewhereinblog.net/`).

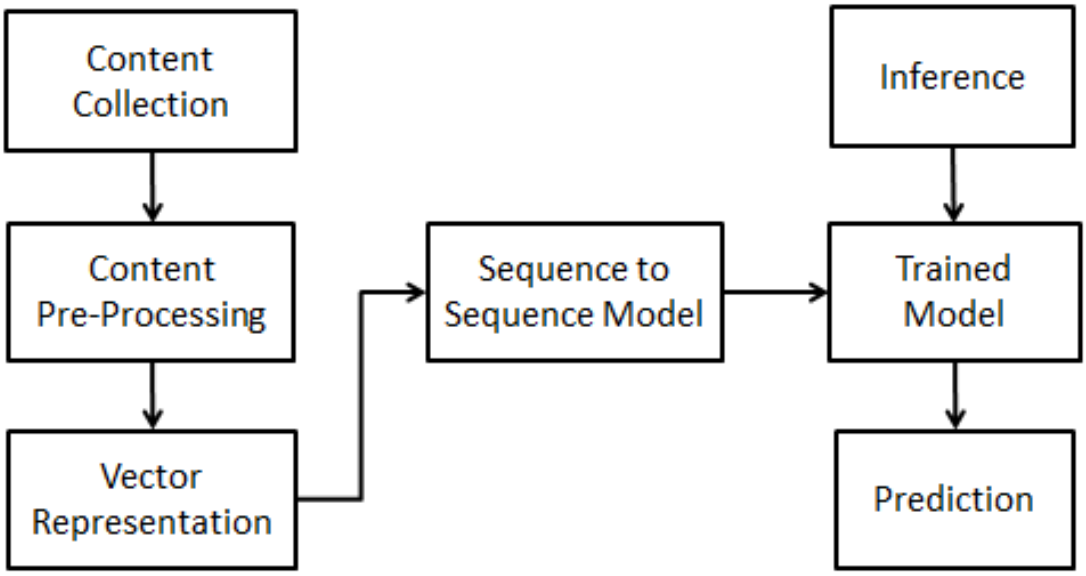


Figure 3.1: System diagram for the methodology used in this paper.

3.2 Content Pre-Processing

After collection of the Bangla sentences, the next step was the pre-processing of the sentences. In this step, for each of the sentences in the input dataset, we considered them as correct sentences. After that, we have used random perturbation to introduce noise in each of the sentences to produce incorrect sentences. Three types of errors were introduced in each sentence using a controlled randomized procedure:

1. Auto-complete: A position was randomly selected in the given correct sentence and the sentence was divided into two parts. The first part was considered as the input sentence and the last part was considered as the output or auto-complete part.

2. Wrong Arrangement: Two words in the sentence were selected randomly and the positions were swapped to introduce mis-arrangement error.

3. Missing Words: A word in a random position was selected in the sentence and was deleted to introduce missing word error.

Thus all the sentences collected were converted to sentence pairs. Following this procedure, we created a massive dataset with 5 million input-output pairs. Distribution

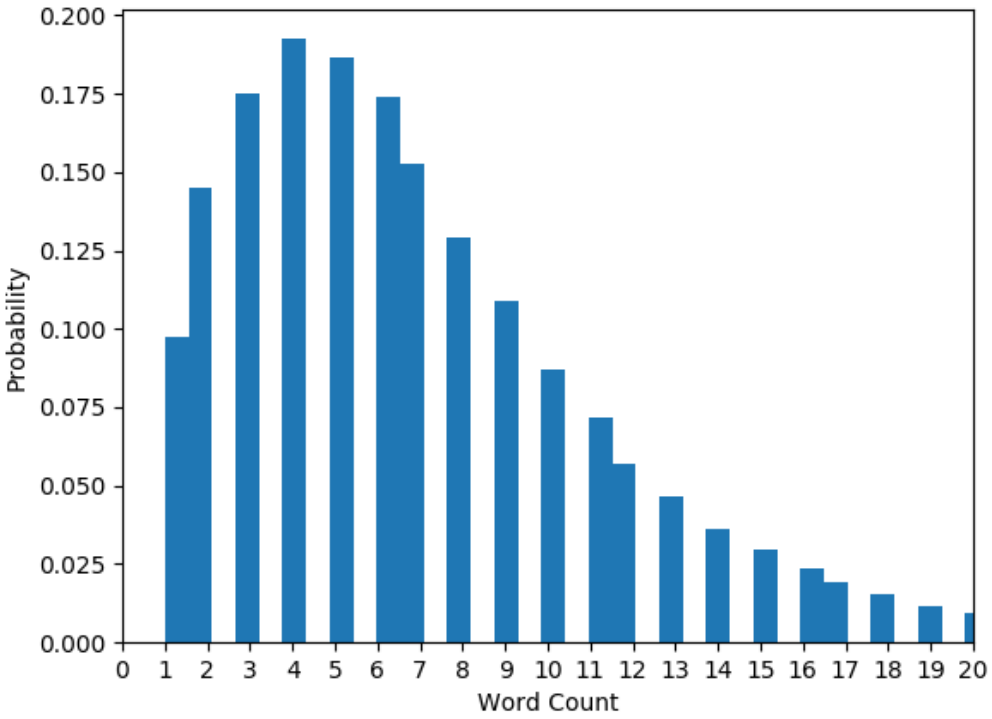


Figure 3.2: Histogram of word counts or sentence length for each Bangla sentence collected in the dataset.

of the errors in these pairs of sentences are shown in Fig.3.3 as total percentages. A few example input-output sentence pairs are shown in Fig.3.5.

3.3 Word Embeddings

In this step, we tokenized the sentences and constructed the vocabulary. For each word in the vocabulary, frequencies were then calculated for the occurrences of them in the whole dataset. After the creation of the vocabulary, we have replaced the rare words of words with the symbol 'UNK' that stands for 'UNKNOWN'. We considered a word with frequency less of equal to 3 as rare word. All the occurrences of numbers were replaced by the token 'NUM'. Fig.3.4(a) shows a toy example dataset with 3 sentences and vocabulary of size 6. Fig.3.4(c) shows the count of each word in the vocabulary constructed from the input sentences.

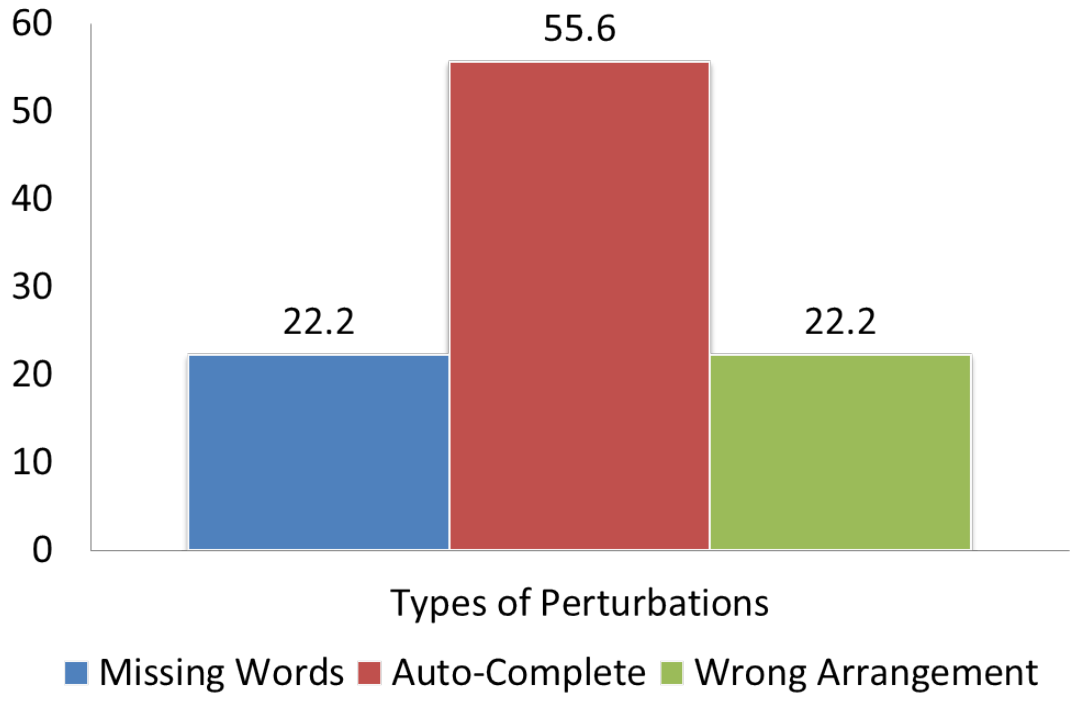


Figure 3.3: Percentage of sentence pair dataset for each type of perturbations generated in the randomized noise generation technique.

After the vocabulary has been created and the frequencies has been calculated, each sentence pair in the pre-processed dataset were replaced by the word embeddings created from the vocabulary. Each word in the the vocabulary including the UNK token were numbered using a real number system given to each of the words as in sorted order of frequency. Fig.3.4(b) shows the vector encodings of the input sentences in the toy example.

3.4 Sequence to Sequence Model

After we have successfully created the dataset and converted them using word embeddings into a vector suitable for neural network, we have to train the model using a sequence to sequence model. In our proposed method, we have used encode-decoder architecture. Our encoder is a Bidirectional Dynamic Recurrent Neural Network (BDRNN) [22]. However, the decoder is a custom recurrent neural network that allows more control. Attention mechanism has been used to communicate between encoder and de-

Input Sentences
আমি ভাত খাই
আমি খেলা দেখি
আমি টিয়া পাখি দেখি

(a)

Vector Representation
1 3 4
1 5 2
1 0 6 2

(b)

Vocabulary	UNK	আমি	দেখি	ভাত	খাই	খেলা	পাখি
Count	UNK	3	2	1	1	1	1
Real Num.	0	1	2	3	4	5	6

(c)

Figure 3.4: A toy example of Word2Vec representation of input sentences: (a) input sentences in Bangla, (b) output Word2Vec Encoding and (c) assignment of real numbers to each words based on the frequency of occurrence.

coder. For each of the Used RNNs Long Short Term Memory (LSTM) cell were being used. The rest of this section describes each of the individual components and other configurations used in the proposed method.

3.5 Recurrent Neural Network

In recurrent neural networks [7], outputs are fed into the input to the next step and a loop is created. Thus networks are able to predict at any time of a sequence based on the inputs and predictions made in the previous steps and thus very much suitable to sequences and lists. An unfolded recurrent neural network is basically equivalent to a series of feed forward networks forwarded information from one to another. RNN and its equivalent feed forward network is shown in Fig.3.6. In this paper, we have used bi

Sl	Perturbation	Incorrect sentence	Corrected sentence
1	Wrong Arrangement	মেসিকে বাঁচানোর জন্য বোধহয় পেনাল্টি রোনাল্ডো মিস করেছে ইরানের সঙ্গে	মেসিকে বাঁচানোর জন্য বোধহয় রোনাল্ডো পেনাল্টি মিস করেছে ইরানের সঙ্গে
2	Autocomplete	মুখরিত স্বপ্নের	মুখরিত স্বপ্নের ইতিহাস
3	Missing word	চায়নার এশিয়ান দেশগুলোর ব্যাপারে জানে	চায়নার মতো এশিয়ান দেশগুলোর ব্যাপারে জানে

Figure 3.5: Input-Output sentece pairs generated in the pre-processing and used in the inference model.

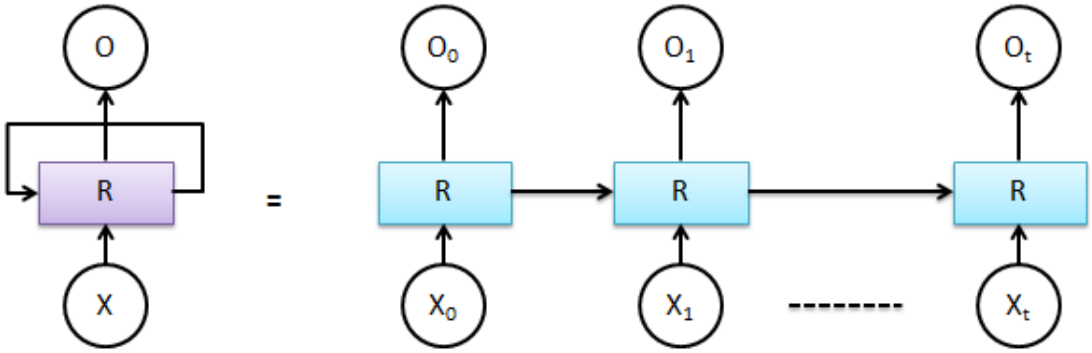


Figure 3.6: RNN and its feed-forward equivalent network.

3.6 Long Short Term Memory (LSTM) Cell

In natural language processing tasks as in auto-completion of sentences, given any word, suppose we are trying to predict the last word. We need only to look at the recent information in the network for the prediction task. However, if the context on which the prediction is dependent then recurrent networks required to have long short term memory cells [23]. A typical LSTM cell is shown in Fig.3.7.

3.7 Encoder Decoder Architecture

We used an encoder-decoder architecture with attention mechanism. The block diagram of our architecture is shown in Fig. 3.8. Note that, in the case of natural language processing and specially sentence correction, the input sentences are of various lengths. Moreover, the output might also be of arbitrary length and independent of the input

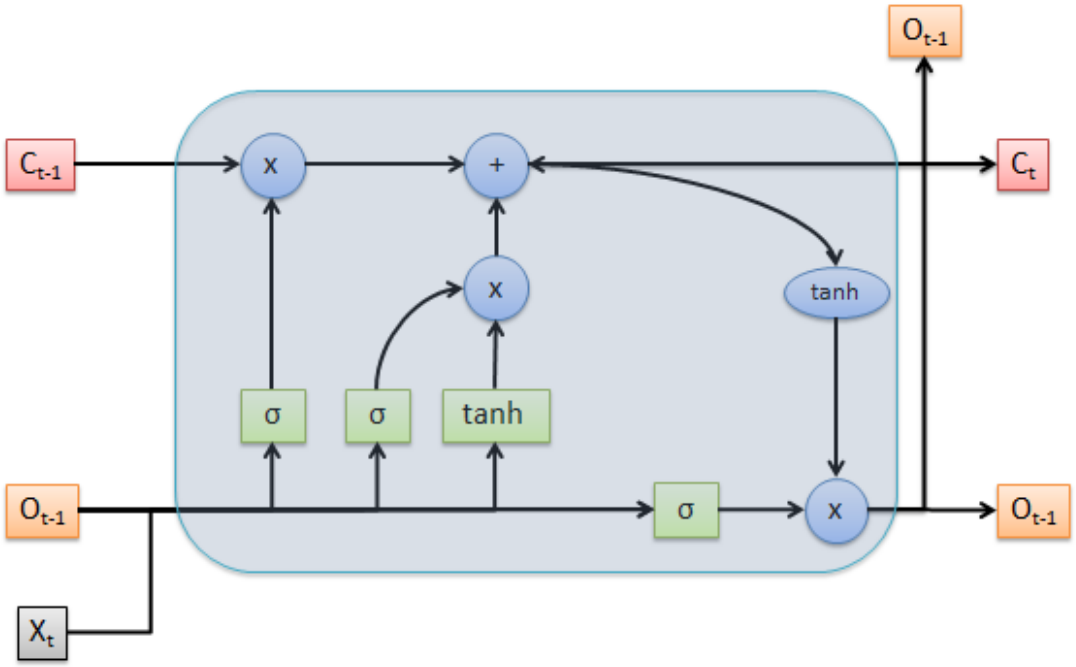


Figure 3.7: A typical LSTM cell.

sentence (in auto-completion, missing words, etc).

In the encoding phase, all the words embedded into vectors are fed into the bidirectional RNN with LSTM cells. The data is transfered from left to right to each cell and words are learned based on the input in the current state and the previous states. Encoder generates output and a hidden state, after which the decoder collects the hidden state from the encoder and generates output or words for the corrected or completed sentence sequentially.

Within this basic encoder-decoder scheme, attention mechanism [21] uses a context vector in between the encoder and the decoder. It collects output of all the units and captures the model of the language by calculating distribution of the words from a global point of view. Thus the decoder is able to generate words based on distant context hidden in the input sequence of words and able to work for various lengths of input sentences.

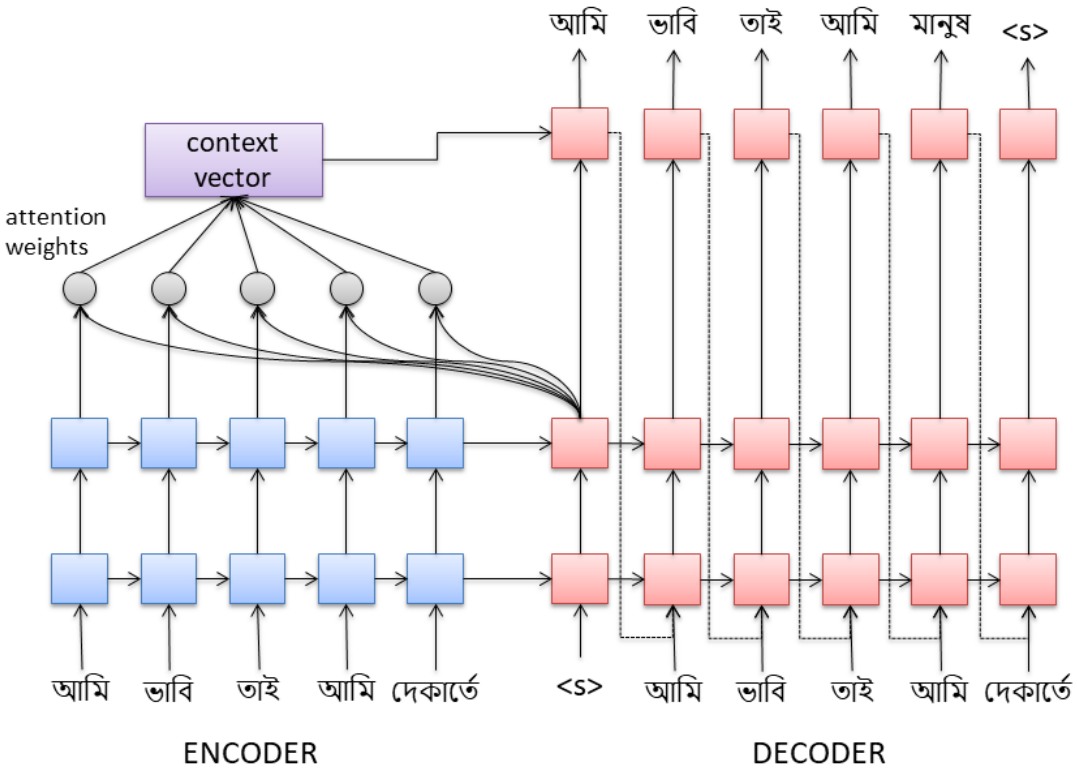


Figure 3.8: Encoder Decoder Architecture implemented with Recurrent Neural Networks with with LSTM cells with attention mechanism.

Chapter 4

Experimental Analysis

All experiments done in this research was run on a Machine with Intel Core i7-7700 3.60 GHz CPU, equipped with 32 GB physical memory with running operating system Ubuntu 14.4 and a NVIDIA Titan Xp GPU of 12GB. All the programs were implemented using Python 3.5 and TensorFlow library. We have used input embedding size of 20 and number of encoder hidden units were kept 128. As the length of the output varies, we used twice as many decoder units in the model.

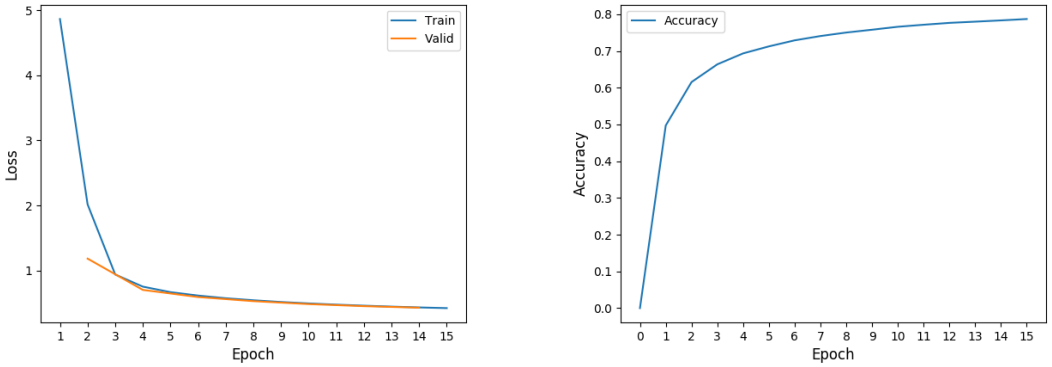
4.1 Training of Data

As the dataset built for this proposed model was large enough (5 million sentence pairs), we followed the general recommendation for deep learning [24] while partitioning the data for experiments. The dataset was randomly shuffled. After shuffling, we have divided the original dataset into three parts.

1. Training Set: It contained 95% of the dataset.
2. Validation Set: It contained 4.5% of the dataset.
3. Test Set: It contained 0.5% of the dataset.

We trained our sequence to sequence model using batches. In each epoch we have used 4000 batches and with a batch size of 512. The total number of epochs was 15. In case of validation, 200 batches were processed. However, this parameter had a value of 40 in the case of test set. The loss function that was used for training and validation is softmax cross entropy.

We have used inference model for testing. For each of the sentence pairs in the dataset (validation or test), the input sentence was fed to the encoder of the network. The output from the decoder was then converted to human readable format or string using the inference model. In Fig. 4.1(a), we plot the loss function in both training and validation phase against the training epochs. Note that, our method is not overfitting the dataset as the validation loss function in almost similar to the training loss function and the model converges after some point. Based on this we have taken the model after the 15 training epochs, since after that the model starts to overfit the data and the loss function diminishes quickly.



(a) (b)

Figure 4.1: Plot of (a) loss function against epochs for train set and validation set and (b) accuracy against epochs on test set.

After training, we have stored the best model that does not overfit and tested the accuracy of the model. Fig. 4.1(b) shows the plot of accuracy on the test set for models learned at different epochs. Note that, the best accuracy achieved by our model is 79% on the test set.

4.2 Benefits of GPU Implementation

We have observed the benefits of training using GPU processor. At first, we started our training with only CPU model and used 128 instances per batch, 1000 batches per epoch and 15 epochs in total. We were able to train only upto half million of the data which is only one tenth of the full dataset. This particular dataset had a vocabulary of

size 6000. It took 11 hours to train this model and beyond this, the CPU model was not able to train due to memory shortage.

However, when trained with GPU processor, the vocabulary size was 28000 and the total instances in the dataset were five million, ten times compared to the CPU model and was able to train that model only in 8 hours using 15 epochs, 4000 batches per epoch and 512 instances per batch. We could easily observe the speed up achieved by the GPU implementation.

4.3 Availability of Materials

It was one of the challenges for us to use any of the datasets previously constructed for various natural language processing task since they are either too small in size to be suitable for training a deep learning network or they are unavailable. To overcome this, we have constructed our own dataset and made this available via online repository. We have also failed to compare our method with any of the previous methods due to the unavailability of their methods or codes. However, we make sure that our results are reproducible and re-usable by other researchers and made our code and implementations available online. All the materials used as data and methods implemented as in code are freely available from: `https://github.com/mrscp/bangla-sentence-correction`.

We have also implemented an online web application for the general users for use and to demonstrate the usability of our method. This web application takes an input sentence from the user as input which is supposedly incorrect and produces the suggestion for a correct sentence. Our web application is available to use from: `http://brl.uiu.ac.bd/s2s`. A screenshot of the web application is given in Fig. 4.2. The website has a very simple interface and also provides a few invalid sentences in human readable format that was used in the testing of the system and could be used by the general users.

Figure 4.2: Screen shot of the web application implemented based on the proposed model in this paper.

Chapter 5

Conclusion

In this paper, we have proposed an encoder-decoder architecture of sequence-to-sequence of deep neural network for Bangla sentence correction and auto completion. Here, the encoder is a bidirectional dynamic recurrent neural network with long short term memory (LSTM) cell. Decoder is a raw recurrent neural network, where we created that is more primitive version of dynamic recurrent neural network that provides more direct access to the input each iteration, it also provides when to start and when to stop reading the sequence and what to emit for the output. It shows how the attention mechanism works in the sequence-to-sequence model. After training the model with the dataset, our model is capable of producing correct sentence and complete incomplete Bangla sentence. Our proposed model achieved 78% accuracy of correcting three kind of error and auto completion. Those are invalid arrange of words in a sentence, missing word in a sentence and auto completion. We have made our methods and data available for the researchers and also implemented an online tool for general users.

We want to upgrade to make better experience of using bangla language in written communication through the technology. Currently, our system just working with words in the system, in future will make a system that will be capable of correct spelling of the words too. The basic thing about performance for this model is the dataset. If the dataset is good, the result will be good as well. So we will improve the dataset in the future, and it is the most vital thing, we suggest. Also, we will improve the model as well, natural machine translation (NMT) model of deep networks is becoming the standard over sequence-to-sequence model on language modeling. So will implement this version of deep networks to correct the sentence and the spelling. Moreover, we

could use sequence-to-sequence convolutional gated recurrent encoder gated decoder, proposed in a recent research [10].

Bibliography

[1] S. Dasgupta and V. Ng, "Unsupervised morphological parsing of bengali," *Language Resources and Evaluation*, vol. 40, no. 3-4, pp. 311–330, 2006. 1, 4

[2] P. Sengupta and B. Chaudhuri, "Morphological processing of indian languages for lexical interaction with application to spelling error correction," *Sadhana*, vol. 21, no. 3, pp. 363–380, 1996. 1, 4

[3] B. Kundu, S. Chakraborti, and S. K. Choudhury, "Combining confidence score and mal-rule filters for automatic creation of bangla error corpus: grammar checker perspective," in *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 2012, pp. 462–477. 1, 4

[4] S. C. Bibekananda Kundu, Sutanu Chakraborti, "Complexity guided active learning for bangla grammar correction," in *10th International Conference on Natural Language Processing, ICON*, 2013. 1, 4

[5] B. Kundu, S. Chakraborti, and S. Choudhury, "Nlg approach for bangla grammatical error correction," in *9th International Conference on Natural Language Processing, ICON*. Macmillan Publisher, 2011, pp. 225–230. 1, 4

[6] M. J. Alam, N. Uzzaman, and M. Khan, "N-gram based statistical grammar checker for bangla and english," *Center for Research On Bangla Language Processing. Bangladesh*, 2006. 1, 4

[7] P. Rodriguez, J. Wiles, and J. L. Elman, "A recurrent neural network that learns to count," *Connection Science*, vol. 11, no. 1, pp. 5–40, 1999. 1, 11

[8] S. Poria, E. Cambria, and A. Gelbukh, "Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment

analysis," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 2539–2544. 1, 5

[9] A. Joshi, A. Prabhu, M. Shrivastava, and V. Varma, "Towards sub-word level compositions for sentiment analysis of hindi-english code mixed text," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2482–2491. 1, 5

[10] S. Ghosh and P. O. Kristensson, "Neural networks for text correction and completion in keyboard decoding," *arXiv preprint arXiv:1709.06429*, 2017. 1, 5, 20

[11] C. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 69–78. 1, 5

[12] N. Majumder, S. Poria, A. Gelbukh, and E. Cambria, "Deep learning-based document modeling for personality detection from text," *IEEE Intelligent Systems*, vol. 32, no. 2, pp. 74–79, 2017. 1, 5

[13] T. Bhowmik and S. K. D. Mandal, "Deep neural network based phonological feature extraction for bengali continuous speech," in *Signal and Information Processing (IConSIP), International Conference on.* IEEE, 2016, pp. 1–5. 1, 5

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

[15] M. Islam, "Research on bangla language processing in bangladesh: progress and challenges," in *8th International Language & Development Conference*, 2009, pp. 23–25. 1

[16] M. Karim, *Technical challenges and design issues in bangla language processing.* IGI Global, 2013.

[17] S. Hossain, N. Akter, H. Sarwar, and C. M. Rahman, "Development of a recognizer for bangla text: present status and future challenges," *Character Recognition, Minoru Mori, Janeza Trdine*, vol. 9, no. 51000, pp. 83–112, 2010. 1

[18] F. M. Hasan, N. UzZaman, and M. Khan, "Comparison of different pos tagging techniques (n-gram, hmm and brills tagger) for bangla," in *Advances and innovations in systems, computing sciences and software engineering.* Springer, 2007, pp. 121–126. 4

[19] A. Das and S. Bandyopadhyay, "Sentiwordnet for bangla," *Knowledge Sharing Event-4: Task*, vol. 2, pp. 1–8, 2010. 5

[20] K. Sarkar and S. Chakraborty, "A sentiment analysis system for indian language tweets," in *International Conference on Mining Intelligence and Knowledge Exploration.* Springer, 2015, pp. 694–702. 5

[21] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014. 7, 13

[22] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997. 10

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. 12

[24] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning.* MIT press Cambridge, 2016, vol. 1. 15