DUJASE Vol. 7 (2) 8-15, 2022 (July) https://doi.org/10.3329/dujase.v7i2.65088

Context-based Bengali Next Word Prediction: A Comparative Study of Different Embedding Methods

Mahir Mahbub, Suravi Akhter, Ahmedul Kabir* and Zerina Begum

Institute of Information Technology, University of Dhaka, Dhaka-1000, Bangladesh

**E-mail: kabir@iit.du.ac.bd*

Received on 24 April 2022, Accepted for publication on 12 February 2023

Abstract

Next word prediction is a helpful feature for various typing subsystems. It is also convenient to have suggestions while typing to speed up the writing of digital documents. Therefore, researchers over time have been trying to enhance the capability of such a prediction system. Knowledge regarding the inner meaning of the words along with the contextual understanding of the sequence can be helpful in enhancing the next word prediction capability. Theoretically, these reasonings seem to be very promising. With the advancement of Natural Language Processing (NLP), these reasonings are found to be applicable in real scenarios. NLP techniques like Word embedding and sequential contextual modeling can help us to gain insight into these points. Word embedding can capture various relations among the words and explain their inner knowledge. On the other hand, sequence modeling can capture contextual information. In this paper, we figure out which embedding method works better for Bengali next word prediction. The embeddings we have compared are word2vec skip-gram, word2vec CBOW, fastText skip-gram and fastText CBOW. We have applied them in a deep learning sequential model based on LSTM which was trained on a large corpus of Bengali texts. The results reveal some useful insights about the contextual and sequential information gathering that will help to implement a context-based Bengali next word prediction system.

Keywords: Context-based next word prediction, word embedding, sequence model, word2vec, fastText

1. Introduction

Because of the digitization trend in every aspect of our life, all types of recorded documentation nowadays tend to be in electronic format. Typing in digital devices can be made faster if the computer is able to accurately predict the next word that is intended. This prediction is important in increasing the productivity of the writer by reducing typing time. Furthermore, a next word prediction system can also be useful in detecting spelling errors and other typing errors. It also reduces the gap between the people who are experts in typing with the people who are differently abled or people who are not regular computer users [1].

There have been some works on Bengali next word prediction, most of which are developed using the n-gram method which can not predict context-relevant words [2]. As it is well known that The n-gram technique is useful for sequence prediction in various fields such as text prediction [3], genome prediction [4] etc. But to truly capture the essence of the language, the words should be predicted from the given context. To understand the importance of context, let us consider an example. The context relevant next word of "সে কবিতা" can be "লিখে" or "পড়ে". In this case, when the previous sentence is "রহিম একজন কবি", the relevant word is "লিখে". In the same way, when the previous sentence is " রহিম কবিতা পছন্দ করে", the next word should be "পড়ে". There have been a couple of attempts at capturing this contextual information in Bengali texts.

In [5], the proposed method consists of a GRU model extended on the n-gram model to give the prediction in context. Another hybrid approach of sequential LSTM and n-gram along with trie implementation [6] is proposed for Bengali word completion and sequence prediction.

In this paper, we conduct a comparison of different embedding methods using an LSTM model for next word prediction. Word embedding is the learned representation of words in numerical format. This embedding contains the inner word level knowledge as well as the relational understanding among words. Four models are developed using different embeddings. We designed LSTM models using word2vec skip-gram and continuous bag of words (CBOW) embedding, as well as fastText skip-gram and continuous bag of words(CBOW) embedding. The main contribution of this research is implementing the four-word embeddings mentioned above, training the next word prediction model for word prediction using those embeddings, and finally analyzing empirically the best-fit embedding for next word prediction using the LSTM network. n-gram models are also developed for the purpose of comparison.

2. Preliminaries

2.1 n-gram

n-gram is a frequency-based method that assigns probabilities to the sequences of words. The n-gram is the sequence of N words of a sentence where the N value can be two (called bigram), three (called trigram), and so on. In the following equation 1, p(w|h) refers to the probability of the word w given the previous n words which is denoted as h.

$$p(w|h) = \frac{\text{Count of w}}{\text{Count of h}} \quad (1)$$

Here w is the word that should be after the previous words h. The count of w and h is from our training corpus. For bigram, next word is predicted given one previous word. For example, probability of word "ভাত" after previous word "আমি", is the ratio of count "আমি ভাত" and "আমি". Suppose count of "আমি ভাত" is 6 and total count of "আমি" is 10 then probability of the bigram model is 6/10=0.6. So for every n-gram, previous N-1 words are given in a sentence in sequence and the n-th word is predicted.

2.2 LSTM

From the human perspective, To predict the next word at the end of the sequence, It is important to know what are the words that are already present in the sequence. So past memories along with the context from sequences are needed to predict the next word. Here the recurrent structure of the neural network comes into play. Recurrent Neural Network(RNN) is widely used for purposes like this as it has a chain-like structure. Though in theory, RNN can preserve long-term dependencies but in practice, it is actually not the case. That is because of the major arithmetic operations applied to the memories that are obtained from the previous cell, resulting in a vanishing gradient problem [7]. Suppose there is a sentence "বিড়াল একটি আদুরে প্রাণী, এটি হল একটি গৃহপালিত প্রাণী যাকে দেখলেই", Here the probable predicted next word should be "আদর". This prediction is influenced by the word "আদুরে" in the sequence. RNN cannot grab this type of long-term dependencies in the practical world. To solve this, RNN with Gate structure (LSTM/GRU) [8] [9] can be used instead of classical RNN. The major difference between RNN and Gate-based RNN(LSTM/GRU) is in their repeating module. For the next word prediction, LSTM [10] is used here. LSTM cells use the cell state with which information passes through the cells with some non-major linear interaction. LSTM have abilities to add or remove any information to cell states by letting the information through to the next cell with an option. LSTM uses the "forget gate layer" to decide whether to throw away the information from the previous layer. Forget gate equation is shown in equation 2. Here W_f is the weighted matrix and b_f is the bias, $h_t{-}1$ is the previous hidden state of the forget gate f_t.

$$f_t = \sigma\,(W_f\,[h_t{-}1,\, x_t] + b_f) \tag{2}$$

"Input gate layer" helps to decide which values of new information should be picked for updating the cell state. "Input gate layer" is shown in equation 3. Here W_i is the weighted matrix and b_i is the bias, $h_t{-}1$ is the input from the previous hidden state of input gate f_i. Overall, the gates are sigmoid units that receive activation from both the hidden layer from the previous time step and the current input x_t.

$$i_t = \sigma\,(W_i\,[h_t{-}1,\, x_t] + b_i) \tag{3}$$

A "tanh layer" is used for generating new candidate values denoted as C_t for the present cell which is shown in equation 4. Here W_c is the weighted matrix and b_c is the bias, $h_t{-}1$ is the input from the previous hidden state.

$$C_t = tanh\,(W_c\,[h_t{-}1,\, x_t] + b_c) \tag{4}$$

Finally values from "Input gate layer" and "tanh layer" are combined to create an update to the state.

2.3 Embedding

Distributed word representation often proved to be helpful while understanding the words and their inner properties. Though those embedding can be divided into two classes based on their context. word2vec and fastText are the embeddings that resulted from the non-contextual scenario. Whereas contextual embedding like ELMO, Bert trained on sequence-level semantics by considering the sequence of words which can result in different representations for polysemous words. Here non-contextual embedding like word2vec and fastText with their different training variants like CBOW and skip-gram are used for word embedding.

1) *CBOW:* CBOW tries to predict the target word given the context words from the contextual window of size T, while considering the n history words and m future words where $T = n + m$. The CBOW architecture, which is shown in Figure 1, is similar to Feedforward Neural Net Language Model (NNLM) [11], where the hidden layers are discarded, and the projection layer is shared for all words in the vocabulary [12]. It tries to maximize the predicting probability of the center word, $p\,(w_c \mid w_t)$, where $t \in T$, w_c represents the center word and w_t represents the neighboring words. Because of its architecture CBOW learns better syntactic relationships between words.

2) *Skip-gram:* Architecture of CBOW model, which is shown in Figure 2, is like the CBOW model except that it predicts the context words based on the current word which is opposite to the CBOW's methodology. It tries to predict the history words and future words from the context windows.

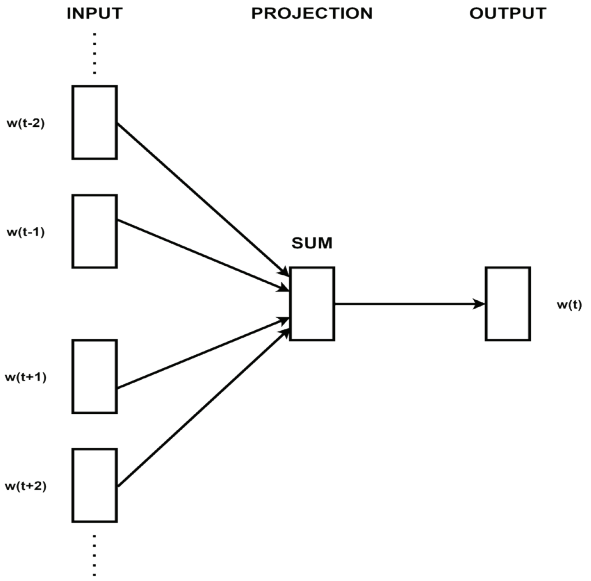

Fig. 1. CBOW Architecture.

This model also discards the hidden layer. The predictive words in the training set are sampled based on the distance from the input word [12]. skipgram tries to maximize the neighboring word predicting probability from the given center word, p (w_1, w_2, …, w_{n-1}, $w_n \mid w_c$) where w_1, w_2, … , w_{n-1}, w_n are the neighboring words and w_c is the given center word. In contrast to the architecture of the CBOW model, the input is one word and the output is a set of words that are considered the neighbors of the given center word. Skip-gram captures better semantic relationships.

3) Word2vec, fast Text and Subword Model: The word2vec and fastText both use the skip-gram and the CBOW architecture. But fastText introduces an additional subword model [13]. To extract the internal structure of the word, the word w is divided into bags of ngrams of itself. The n-gram bag also includes the main word w itself. The embedding of the word w is the sum of the embedding of the elements in the corresponding bag of words w [14]. In skip-gram model, the probability of word w_c given surrounding word w_t is resembled in equation 5,

$$p(w_c|w_t) = \frac{e^{s(w_t,w_c)}}{\sum_{j=1}^{W} e^{s(w_t,j)}} \quad (5)$$

$$s(w,c) = \sum_{g \in G_w} z_g^T v_c \quad (6)$$

In equation 6, the scoring function, which is denoted as s, is the dot product of two-word embedding in the original model. Whereas, in fastText, s is denoted as the sum of the vector representations of n-grams of the word w_c [13].

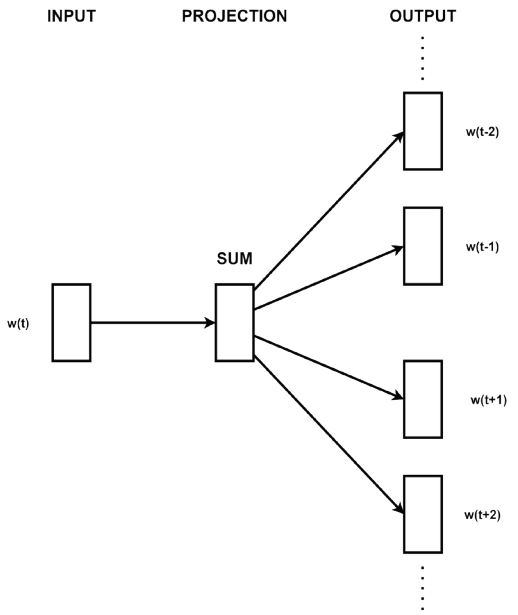

Fig. 2. Skip-gram Architecture

where G_w is the set of n-grams of word w and z_g is the vector representation for every n-gram g. It is also same for fastText CBOW architecture. In this paper, the embedding length is taken as 300. It is the standard length of the vector to capture the necessary features in the embedding. The embedding size less than or bigger than this can lead to missing the important features or overestimating the features.

2.4 Next Word Prediction

The next word prediction works based on contextual knowledge of past words. In n-gram, the next word is predicted based on the availability of the n-gram on the training set. n-gram models do not consider linguistic knowledge while predicting the next word. It gives the same importance to all the past words in the sequence. By linguistic knowledge, concepts such as inner word level knowledge, inter-relational understanding of words (such as semantic and syntactical knowledge) along with the contextual understanding of the sequence are meant. In the LSTM-based model that is used in this paper, the knowledge of linguistic instincts is taken into consideration by injecting word embedding in the embedding layer and also through the LSTM layers. Also, positional importance is learned in the model while considering the word's position in the sentence. So in terms of the context of sequence in a sentence, the LSTM model based on embedding should work well in comparison to the n-gram model and plain LSTM-based modeling because those models can not provide all the characteristics that are mentioned above.

3. Literature Review

Next word prediction is a Natural Language Processing (NLP) task which is one of the important topics regarding NLP research. n-gram method is the early approach for the completion of a sentence [15]. This is developed for English. Another approach [16] is proposed where a novel relevance and enhanced Deng entropy-based Dempster's combination rule is proposed for next word prediction. Actually, for English, there are numerous efforts for accurate word prediction [17]. However, there are only a small number of works in the Bengali Language for next word prediction. One of the major works in this regard uses the n-gram method [2]. They use unigram, bigram, trigram, back-off propagation, and deleted interpolation which are the various variants of n-gram. the computational cost of these approaches is very high as they count N-before state as well as N-after state frequencies.

Naïve Bayes is also used for next word prediction for English language [18]. This model states that its variables can be divided into cause and effect behaviors. Thereby, it can be assumed that the effects are conditionally independent between themselves, which reduces the computational cost of the model over n-gram. However, it suffers from the same problem as n-grams. It cannot reliably predict context-relevant words.

To solve this problem associated with n-gram and Naïve Bayes, Hidden Markov Model is introduced for next word prediction, where the next word depends only on its previous

n-states [19]. An explanation of the process is given below. Suppose three sentences - I like Science, I like Photography, I love Mathematics. All the unique words from the sentences above ('I', 'like', 'love', 'Photography', 'Science' and 'Mathematics') could form different states. 'like' appeared two times after 'I', so the more probable word after 'I' is 'like'. Words are predicted using only their previous N-states, not based on the context.

Mikolov et. al [20] implemented next word prediction of sequential data using Recurrent Neural Network for the English Language. Zhou et. al [21] implemented text classification and predictions on sentence representations by using a combined approach of CNN and LSTM.

In [22], the authors propose both a word completion and sentence completion using trie and sequential LSTM model. Their proposed trie model is supposed to be dynamic. As the vocabulary increases, it checks the word prefix, if exists in the model it suggests the probable word otherwise it approves the new word with some spelling check. Sequential LSTM is used to detect the context and complete the sequence of sentences given some previous words. They show that their model has a better suggestion and word completion capability rather than n-gram. They did not specify any word embedding model that is used in their language model.

There are some works published for next word prediction in several other languages apart from English. For example, an n-gram based [23] and an RNN based [24] next word prediction model for the Assamese language has been proposed. In [25], some sequence prediction models were explored to evaluate the performance of the next word prediction for Hindi. The same analogy is applied to Ukrainian language [26] to evaluate next word prediction models performance for existing sequence models.

In [27], the authors experimentally fine-tune the word2vec embedding parameter to get better performance for Bengali word embedding. The authors claim that 300 dimensions with 4 window sizes for word2vec skip-gram model performs best for robust vector representation of the Bengali language. They mainly use newspaper datasets where the dataset can be categorized into economy, international, entertainment, sports, and state. Their finetuned word2vec skip-gram model achieves 90% purity in word clustering.

In [28], they applied analysis on word2vec and fastText embedding for Bengali dataset. There have been many researches on finding appropriate word clustering and ngram models are one of them. Now, with the improvement of deep learning methods, dynamic word clustering models are preferred because they reduce processing time and improve memory efficiency. Word2Vec and fastText are two popular dynamic word embedding methods and the authors use these two embeddings. They found that fastText embedding performs better than the remaining embedding in word clustering. The reason is that fastText can give a cluster for an unknown word where other methods can not.

In [29], the researchers proposed a language model which utilizes GRU architecture along with the n-gram for predicting next word. Next word suggestion makes it easier to complete user-user communication. The result is promising because GRU can capture context for the output prediction. They use newspapers (BBC, Prothom Alo) and academic datasets for their experiment and found out that LSTM performs best in word prediction and sentence completion when the previous 5 words are given.

In [30], sentence generation from a sequence of words is useful in machine translation, speech recognition, image captioning, language identification, video captioning, and much more. In this paper, the author develops a text generation method using a deep learning approach and taking advantage of Long Short-term Memory (LSTM) for capturing context. But their dataset was so small only contain 4500 sentences. So the performance will decrease in unseen data as the model is trained using a small set of data.

4. Experimental Setup

4.1 Corpus Description

Our corpus combines a corpus containing Bengali newspaper articles (*Prothom Alo*, *Kalerkontho*, *Ittefaq*) and Bengali Wikipedia data. The data is collected by crawling and by manual approach, Then the collected data are preprocessed for further use. Table 1 shows the proportions of data we collected from different sources.

Table 1: Details of the datasets used for corpus building.

Data	Proportion (%)
Bangla_newspaper_article	65.36
Bangla Wikipedia	34.64

The total number of sentences in the corpus is 10,340,273, which contains 8,928,472 unique words. We use 25,000 sentences in the LSTM model for next word prediction in 4 different models. We split 90% for training and 10% for testing. We used filters to remove special characters & symbols and English words. We use the whole corpus for word embedding and a portion of the data mentioned above for next word prediction.

4.2 Implementation Details

A neural model that is used here, is preceded by an embedding layer and an LSTM [8] layer. the prediction layer uses adaptive softmax [31] same as the unique number of words in the dataset used for next word prediction. The LSTM model architecture is as shown in Figure 3.

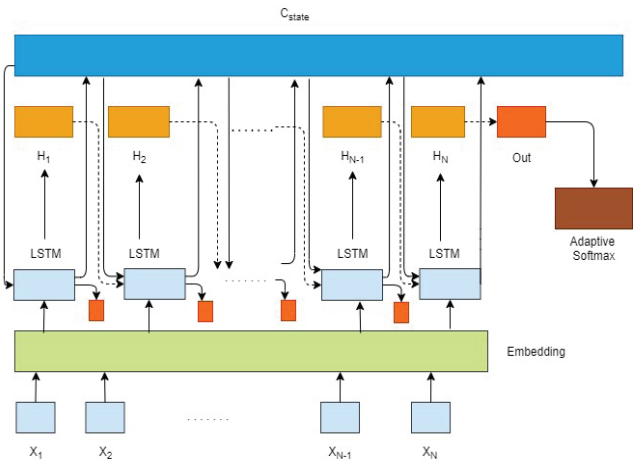

Fig. 3. Next word prediction Model Architecture.

The embedding layer transforms the words to the corresponding representative vectors of float type using the word-vector mapping from the embedding and passes them to the LSTM layer. The LSTM layer gains knowledge from previous words in the window whose size is set to 5. Then the LSTM layer pass the accumulated knowledge to the prediction layer. Adaptive softmax layer [31] transforms the knowledge vector to the probabilistic distribution representative vector which is considered to be the prediction for next word. The words with the higher probability are considered to be the more probable predicted next word in the sequence.

In the embedding layer, different types of word embeddings are used. word2vec and fastText with their two training variants named CBOW and skip-gram, which are trained from the embedding dataset described above.

When training the Word2vec and fastText model, the window size is set to the length of 5. Embedding dimension for Word2vec, fastText which is injected to the nontrainable embedding layer's embedding size is defined to the length of 300. Alongside this, the bigram, trigram, quadgram and pentagram-based next word prediction model is also implemented [15] to compare them with the other models implemented above. The evaluation results are described in detail in the later section.

5. Results and Discussion

The performance of the models has been evaluated using accuracy and perplexity. The Word2vec and fastText embeddings, both with their two variants named skip-gram and CBOW are used in the embedding layer of the next word prediction model. Sequentially, four n-gram based models are implemented for the same purpose of predicting the next word. In Table 2, N denotes the number of predictions to take into account during calculating the accuracy. When the test label matched any of the N predictions, the result is taken as positive.

Table 2: Prediction accuracy of different language model.

Language Model	*N*=1	*N*=2	*N*=3	*N*=4	*N*=5
bigram	2.33	3.60	4.38	5.16	5.56
trigram	17.88	22.39	24.60	25.72	26.47
quadgram	33.28	38.56	40.85	42.03	42.80
pentagram	44.81	50.31	53.64	55.85	58.62
w2v CBOW	35.61	43.87	49.05	52.90	55.30
w2v skip-gram	**52.16**	**63.56**	**70.66**	**75.81**	**79.72**
fast. CBOW	45.14	47.75	54.96	57.95	59.94
fast. skip-gram	46.50	51.91	56.69	59.80	62.97

The CBOW essentially captures syntactical relation better than the skip-gram [32]. On the contrary, skipgram obtains better semantic relations than CBOW. Here, word2vec with skip-gram works better than word2vec with CBOW. fastText works better with unseen words as the construction of the embedding is also done on the character level using n-gram construction. Here the minimum size of the n-gram was set to 3 and the maximum size of the n-gram was set to 6. Though It has been shown that this character-level n-gram subword embedding gives better intuition for syntactical relation in contrast to the semantic relation because of the subword based grammatical representation provided in fastText [13] training. So fastText CBOW shows better results than word2vec CBOW but significantly less accuracy for fastText skip-gram than the word2vec skip-gram model. Therefore because the n-gram models do not use the embedding-based unsupervised features, they do not work well in comparison to the embedding-based LSTM models. This common type of intuition does not always match all languages but Bengali because Bengali is a grammatically and syntactically rich language and strictly typed. For Bengali, certain swaps of words within a sentence can change the meaning of the expression. The above model shows the highest obtained accuracy for all N where $N \in \{1, 2, 3, 4, 5\}$.

The perplexity of a model defines how well the probability model predicts. In equation 7, $PP(p)$ is defined as the perplexity of the discrete probability distribution p and $H(p)$ is the entropy of the distribution p.

$$PP(p) = 2^{H(p)} \tag{7}$$

From the Natural Language perspective, perplexity can be defined as the inverse probability of a test set which is normalized by the total number of words. Equation 8 is equivalent to the defined statement above, where N is the number of words in test set, and w represents the word.

$$PP(W) = \sqrt[N]{\frac{1}{P(w_1 w_2 ... w_N)}} \tag{8}$$

By considering our next word prediction model, the perplexity is defined in equation 9. Here w_i is the word for which the probability is calculated depending on the other words, $w_1 ... w_i$–1.

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1...w_{i-1})}} \quad (9)$$

In Table 3, the perplexity of next word prediction model for different word embedding models are given.

Table 3: Perplexity of different language model.

Language Model	w2v CBOW	w2v skipgram	fast. CBOW	fast. skipgram
Perplexity	87.82	**29.32**	87.02	61.23

The lower perplexity value indicates that the model did well in predicting the result. Here, the LSTM model based on word2vec skip-gram embedding shows the lowest value among all four LSTM models. The perplexity values show consistency with the accuracy results.

In Table 4, some intuitive examples are presented regarding next word prediction from the constructed models that are described above. For each of our embeddings, the top 5 words predicted as the next words are shown. The underlined words are the actual label matched with the test dataset label. It can be observed from the table that the predicted words from LSTM models based on CBOW embedding puts more emphasis on the syntactical properties. For example, connector words like সঙ্গে,

Table 4: Next word prediction examples.

sentence	label	w2v CBOW	w2v skipgram	fast Text CBOW	fast Text skipgram
তাদের কক্সবাজার সদর হাসপাতালে	চিকিৎসা	ভিতর্, চিকিৎসাধীন, নিয়ে, ছাতক, এলে	চিকিৎসাধীন, এলাকার, জরুরী, বিকালে, আহত	চিকিৎসাধীন, চিকিৎসা, নিয়ে, ভিতর, এলে	চিকিৎসাধীন, চিকিৎসা, হাসপাতালে, যাবার, ডাক্তার
ঢাকায় রেকর্ড	করা	করা, করে, পরিমাণ, হয়েছে, করেছে	পরিমাণ, তাপমাত্রা, করা, সূচকের, অনুযায়ী	করা, করেন, হলে, দিতে, হচ্ছিল	পরিমাণ, থেকে, করা, উদ্ধার, তাপমাত্রা
ডেঙ্গু ভাইরাস মোকাবিলায় সকলকে অবশ্যই ঐক্যবদ্ধভাভাবে	কাজ	কাজ, ব্যবস্থা, এবং, দিলে, করে	কাজ, থাকায়, শক্তভাবে, সময়, এলাকার	কাজ, করেছে, হলে, সঙ্গে, এবং	কাজ, এগিয়ে, প্রতিরোধ, এগিয়ে, এলাকা
অতঃপর সে বাসায় যায়, সেখানে গিয়ে সে	তার	তার, স্ত্রী, নিজের, আপন, ও	স্ত্রী, ছেলেমেয়ে, সন্তান, দেখতে, ভিতরে	নিজ, তার, মা, মেয়ে,ও	স্বামী, আত্মীয়, মেয়ে, দরজা, সন্তান
অনুষ্ঠানে কবির পুত্রবধু কাজী নজরুল ইসলামের স্মৃতিচারণা করেন এবং কবির	লেখা	জন্য, উদ্দেশ্যে, স্ত্রী, তৃপ্ত, জীবন	রচিত, লেখা, স্বরণে, আত্মার, কবরে	জন্য, লেখা, বিদেহী, নিজ, উইল	রচিত, উপন্যাস, কবর, স্বরণে, গান

এবং etc. are output as predictions. On the other hand, the prediction from LSTM models based on skip-gram embedding shows emphasis on semantic properties. For example, পরমিাণ, এলাকার etc. are predicted. FastText-based LSTM models can predict words that do not belong to our vocabulary because of the character level n-gram based embedding. For example, word এলাকা was not found in the embedding corpus which is the root word of এলাকার. FastText constructed embedding for this word by using the character level n-gram. In the 4th and 5th examples in Table IV, the context of the sentences is maintained while predicting the next word. The first part of the 4th sentence is talking about someone going to his/her home. the two parts of the sentences are separated by commas. The second part of the same sentence is talking about some predictive event. Here the prediction results are the most relevant to the family-related topic which indicates consistency with the first part of the sentence. In the 5th sentence, the predictions are also contextually related to a good degree. So from this discussion, it can be concluded that LSTM-based next word prediction approaches with embedding preserve the context information while predicting the next word, which is an improvement from the traditional n-gram based approaches.

6. Conclusion

In this paper, the four different word embedding models are constructed upon our own corpus. Then they are applied individually upon an LSTM sequence prediction model structure to make a comparison between them for the purpose of Bangla next word prediction. Additionally, bigram, trigram, quadgram, pentagram models are also constructed on the same corpus that is used for the construction of LSTM model. From the result, It can be seen that the word embedding models work better than the n-gram models. Therefore, among the word-embedding based LSTM models, the model constructed with word2vec skip-gram embedding shows the highest obtained accuracy. So from the findings, It can be asserted that the custom word embedding can provide useful information to the sequence model about word level understandings while predicting the Bangla next word in the sequence. The contextual understanding is also enhanced in our model while predicting the next word which was provided by the LSTM model's architecture. Finally, it can be said that context-based Bengali next word prediction models perform better than other noncontextual models.

The next word prediction can be used in many ways. It can enhance the writing capability of a user or differentlyabled people. Therefore the next word prediction model can be used as a masked model which can be used in the construction of other tools, like spell checker [33] [34]. So we hope our findings will help future researchers and accelerate advanced research which uses the next wordbased masked modeling. In the future, this work can be extended by using a larger and more diverse corpus. Also, the efficacy of the next-word prediction models can be tested by applying them to some practical NLP tasks.

References

1. N. Garay-Vitoria and J. Gonzalez-Abascal, ``Intelligent word-prediction to enhance text input rate (a syntactic analysis-based word-prediction aid for people with severe motor and speech disability),'' in Proceedings of the 2nd international conference on Intelligent user interfaces, pp. 241—244, 1997.
2. M. Haque, M. Habib, M. Rahman et al., ``Automated word prediction in bangla language using stochastic language models,'' arXiv preprint arXiv:1602.07803, 2016.
3. N. Garay-Vitoria and J. Abascal, ``Text prediction systems: a survey,'' Universal Access in the Information Society, vol. 4, no. 3, pp. 188--203, 2006.
4. T. S. Rani and R. S. Bapi, ``Analysis of n-gram based promoter recognition methods and application to whole genome promoter prediction,'' In silico biology, vol. 9, no. 1, 2, pp. S1--S16, 2009.
5. O. F. Rakib, S. Akter, M. A. Khan, A. K. Das, and K. M. Habibullah, ``Bangla word prediction and sentence completion using gru: an extended version of rnn on n-gram language model,'' in 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI). IEEE, pp. 1--6, 2019,
6. S. Sarker, M. E. Islam, J. R. Saurav, and M. M. H. Nahid, ``Word completion and sequence prediction in bangla language using trie and a hybrid approach of sequential lstm and n-gram,'' in 2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT). IEEE, pp. 162--167, 2020.
7. S. Hochreiter, ``The vanishing gradient problem during learning recurrent neural nets and problem solutions,'' International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, vol. 6, no. 02, pp. 107--116, 1998.
8. S. Hochreiter and J. Schmidhuber, ``Long short-term memory,'' Neural computation, vol. 9, no. 8, pp. 1735--1780, 1997.
9. K. Cho, B. Van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, ``Learning phrase representations using rnn encoder-decoder for statistical machine translation,'' arXiv preprint arXiv:1406.1078, 2014.
10. M. Sundermeyer, R. Schluter, and H. Ney, ``Lstm neural networks for language modeling,'' in Thirteenth annual conference of the international speech communication association, 2012.
11. Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, ``A neural probabilistic language model,'' Journal of machine learning research, vol. 3, no. Feb, pp. 1137--1155, 2003.
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. P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, ``Enriching word vectors with subword information,'' Transactions of the Association for Computational Linguistics, vol. 5, pp. 135--146, 2017.
14. A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, ``Bag of tricks for efficient text classification,'' arXiv preprint arXiv:1607.01759, 2016.
15. S. Bickel, P. Haider, and T. Scheffer, ``Predicting sentences using ngram language models,'' in Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pp. 193--200, 2005.
16. G. L. Prajapati and R. Saha, ``Reeds: Relevance and enhanced entropy based dempster shafer approach for next word prediction using language model,'' Journal of Computational Science, vol. 35, pp. 1-11, 2019.
17. K. Trnka, J. McCaw, D. Yarrington, K. F. McCoy, and C. Pennington, ``User interaction with word prediction: The effects of prediction quality,'' ACM Transactions on Accessible Computing (TACCESS), vol. 1, no. 3, pp. 1--34, 2009.
18. H. X. Goulart, M. D. Tosi, D. S. Goncalves, R. F. Maia, and G. A. Wachs-Lopes, ``Hybrid model for word prediction using naive bayes and latent information,'' arXiv preprint arXiv:1803.00985, 2018.
19. G. Szymanski and Z. Ciota, ``Hidden markov models suitable for text generation,'' in WSEAS International Conference on Signal, Speech and Image Processing (WSEAS ICOSSIP 2002). Citeseer, pp. 3081--3084, 2002,
20. T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, ``Recurrent neural network based language model.'' in Interspeech, vol. 2, no. 3. Makuhari, pp. 1045—1048, 2010,
21. C. Zhou, C. Sun, Z. Liu, and F. Lau, ``A c-lstm neural network for text classification,'' arXiv preprint arXiv:1511.08630, 2015.
22. S. Sarker, M. E. Islam, J. R. Saurav, and M. M. H. Nahid, ``Word completion and sequence prediction in bangla language using trie and a hybrid approach of sequential lstm and n-gram,'' in 2nd International Conference on Advanced Information and Communication Technology (ICAICT). IEEE, 2020, pp. 162—167, 2020.
23. M. Bhuyan and S. Sarma, ``An n-gram based model for predicting of word-formation in assamese language,'' Journal of Information and Optimization Sciences, vol. 40, no. 2, pp. 427--440, 2019.
24. P. P. Barman and A. Boruah, ``A rnn based approach for next word prediction in assamese phonetic transcription,'' Procedia computer science, vol. 143, pp. 117--123, 2018.
25. R. Sharma, N. Goel, N. Aggarwal, P. Kaur, and C. Prakash, ``Next word prediction in hindi using deep learning techniques,'' in 2019 International Conference on Data Science and Engineering (ICDSE). IEEE, pp. 55—60, 2019,
26. K. Shakhovska, I. Dumyn, N. Kryvinska, and M. K. Kagita, ``An approach for a next-word prediction for ukrainian language,'' Wireless Communications and Mobile Computing, vol. 2021, 2021.
27. R. Rahman, ``Robust and consistent estimation of word embedding for bangla language by fine-tuning word2vec model,'' in 2020 23rd International Conference on Computer and Information Technology (ICCIT). IEEE, pp. 1--6, 2020.
28. 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, pp. 1--5, 2018.
29. O. F. Rakib, S. Akter, M. A. Khan, A. K. Das, and K. M. Habibullah, ``Bangla word prediction and sentence completion using gru: an extended version of rnn on n-gram language

model," in 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI). IEEE, pp. 1—6, 2019,

30. M. S. Islam, S. S. S. Mousumi, S. Abujar, and S. A. Hossain, ``Sequenceto-sequence bangla sentence generation with lstm recurrent neural networks,'' Procedia Computer Science, vol. 152, pp. 51--58, 2019.

31. A. Joulin, M. Cisse, D. Grangier, H. Jegou et al., ``Efficient softmax approximation for gpus,'' in International conference on machine learning. PMLR, pp. 1302—1310, 2017,

32. T. Mikolov, K. Chen, G. Corrado, and J. Dean, ``Efficient estimation of word representations in vector space,'' arXiv preprint arXiv:1301.3781, 2013.

33. A. Pal and A. Mustafi, ``Vartani spellcheck--automatic contextsensitive spelling correction of ocr-generated hindi text using bert and levenshtein distance,'' arXiv preprint arXiv:2012.07652, 2020.

34. Y. Hong, X. Yu, N. He, N. Liu, and J. Liu, ``Faspell: A fast, adaptable, simple, powerful chinese spell checker based on dae-decoder paradigm,'' in Proceedings of the 5th Workshop on Noisy Usergenerated Text (W-NUT 2019), pp. 160—169, 2019,