Next Word Prediction in Hindi Using Deep Learning Techniques

Radhika Sharma, Nishtha Goel, Nishita Aggarwal, Prajyot Kaur and Chandra Prakash
Indira Gandhi Delhi Technical University For Women, New Delhi, India 110006
Email: radhika2414@gmail.com, nishthagoel26@gmail.com, naggarwal97@gmail.com,
prajyotkaur@gmail.com, cse.cprakash@gmail.com

***Abstract*—Natural Language Generation (NLG) focuses on the generation of natural, human-interpretable language. This study proposes a novel methodology to predict the next word in a Hindi sentence. By predicting the next word in a sequence, the number of keystrokes of the user can be reduced. Two deep learning techniques namely Long Short Term Memory (LSTM) and Bi-LSTM have been explored for the task of predicting next word and accuracy of 59.46% and 81.07% was observed for LSTM and Bi-LSTM respectively. This approach may be used for various NLG tasks like story auto-completion, sentence autocompletion, etc.**

***Index Terms*—Next Word Prediction, Hindi, LSTM, Bi-LSTM**

I. Introduction

Natural Language Generation is the part of machine learning used to generate natural language from data input to it. It is used for translating data to natural language [1]. Various applications of NLG are text generation, summarization, auto-captioning of images, machine translation, etc. Automatic text generation involves generating new text by applying deep learning techniques on the text given . Next Word Prediction involves predicting the next word/s, which have a high probability of following the given sequence of words.

The model suggests a few words which are most likely to follow the current set of words from which the user can select the word of their choice. This helps in saving keystrokes of the user. This was extended to continue predicting the next few words for a given sequence of words. Word prediction is an elementary part of Natural Language Generation. It has various applications, such as the following :

- Helps in minimizing the keystrokes of users while typing.
- Helps save typing time of users.
- Helps in minimizing spelling mistakes of users. Especially useful for users who are not proficient or non-native to that language.
- Aids the non-native users in learning the language by suggesting new and correct words to them, thus expanding their vocabulary.

Hindi is a widely spoken language in India . Thus next word prediction in Hindi will be helpful to a vast majority of people living in India. The processing of the Hindi language is very difficult as it contains a lot of mantras and symbols. This can result in irrelevant results due to the confusion of spellings. Hence, processing the language at the word level gives better results and is less complex.

978-1-7281-2087-4/19/$31.00 © 2019 IEEE

The paper is organized as follows. Section 2 talks about the literature survey done about Next word prediction & Hindi language. Section 3 discusses the methodology proposed, including the specifications of the dataset used. Section 4 illustrates the results obtained. Section 5 discusses the conclusions drawn and the suggested future work.

II. Literature Survey

Natural Language Generation (NLG) is a systematic and significant approach to produce meaningful text that is understandable by humans. For generating the text, the data is collected from different sources or taken as input from the users. There has been a drastic change in the field of NLP over the past few years [2]. Previously, NLP techniques employed shallow machine learning models, which consisted of handcrafted features and very very time-consuming. Due to the increasing popularity of word embeddings, neural networks have achieved greater success in comparison to traditional machine learning models [3].

The English language is very widely used across the world. Hence, researchers have employed several statistical models and machine learning models at the character level and word level for the generation of text . Various statistical models that have been used for text generation in the English language include word2vec approach, a continuous bag of words, etc. which help in generation of text by creating a vocabulary. One limitation of these approaches is that they generate text without ensuring the syntactic correctness of the sentence. Whereas, Lemmatisation, Latent Semantic Analysis (LSA) [4], Parts of Speech (POS) Tagging [5] generate text that is free from any grammatical errors.

Neural Networks also gained popularity over the traditional methods due to their correctness in generating the text. Neural Networks are inspired by the functioning of the brain. RNN was popularly used for text generation because of their ability to process sequential data. But due to its limitation of vanishing gradients, it is being replaced by other neural networks. LSTM, and other versions of LSTM, i.e., Bi-LSTM, GRU are being popularly used nowadays for generating text in the English language. These models are also being used

for other NLP related tasks like Query auto-completion, story generation, etc.

Although English is the language being used worldwide in India, Hindi is the language used by the majority of the citizens for communication. A greater part of the population has little or no knowledge of Hindi language. Hence, a machine learning model which generates text in the Hindi language would help such citizens to connect with the world digitally.

Hindi is a very morphologically rich language consisting of 35 consonants, 11 vowels and 12 matras [6]. Thus, Hindi is a highly complex language. This makes the pre-processing of Hindi language a complex task . Several problems that are faced during pre-processing of Hindi text are -

- Large character set
  The character set of Hindi language is extremely vast. Along with the characters, it also consists of special symbols (Eg- "." , anuswar, halant, etc. This might decrease the no. of keystrokes saved by the model and lead to a decrease in its accuracy [7].
- Phonetically similar characters
  There are characters in the Hindi language that is near of the same shape and size or characters that sound similar. Thus, if such characters are present in the text, they can give rise to confusion.
- Typographical Variants
  A character can have more than one form of representation. This poses a problem while generating text as many alternate forms of a single character exist.

Research has been done to build several statistical and syntactical models that work at character level to generate text in the Hindi Language. But, the character level approach requires to break the sentence character by character . And, due to the complexity of Hindi Language, this approach is difficult and can give rise to ambiguities and generate irrelevant output. Thus, it is recommended to process the data at the word level, as firstly it is easier to understand and process, and secondly, it generates text with better accuracy.

Bi-gram and tri-gram statistical predictors have been used most commonly till date [7]. They predict the next word based on the previous two and three words, respectively. But, the drawback of these statistical models is that they fail to perform well in case of a large data corpus. Parts of Speech (POS) Tagging is a syntactical approach which has also been used by the researchers. But, this approach cannot handle words which are not present in the vocabulary. Various approaches like K-Nearest Neighbour (KNN) [8], Universal Networking Language (UNL) have been used earlier, which generate Hindi text from given English text corpus.

All the above approaches most commonly use character level architecture, i.e., the model helps in predicting the next character in the sequence. One advantage of character level models is that the size of the vocabulary is small. But such models suffer from the vanishing gradients problem [9]. Also, character level processing of a Hindi corpus is highly complex, and hence, word level architectural models provide better results and are less prone to errors. Models that follow word level architecture take a lesser amount of time to train, and generate more logical results by predicting the next complete word instead of only a character. Although, word level architecture requires to store all the individual words, which means the size of the vocabulary is large, hence more memory is required [10]. But, the accuracy given by word level architecture overcomes this drawback.

III. Methodology

The process starts with the cleaning of the dataset. Then, the dictionary of unique words is created by splitting the words in the dataset and filtering the unique words which constitute to a dictionary. The input file is parsed with the iterator, and unique words are collected. Succeeding, the unique words in the dictionary are mapped to indices. This indicates that the words are not easily processed by neural networks in machine learning, and it is important to map it to the indices, which are easy for neural networks to process.

Set sequence length by dividing the sentence into 6:1 ratio as input x and input y and mapping the first six words in input x. Divide the input file into tensor (dataX) based on the sequence length. Create another tensor (dataY) containing the next words of the sequence, which is in the input file which is the output of the project [11].

Assign probability to the content of dataY using softmax (activation) function. Select next word based on the probability of dataY, which will be the predicted word of the sequence. Since the aim of the process is to predict the next word, the predicted word completes the sequence. Each iteration user will be given three options in the output among which user will have to choose the most appropriate option in context to its application and the next word prediction will be based on the output chosen.

Repeat the process till the sequence is completed, i.e., the sequence of seven words are not completed, and our objective is not completed. Repeat the process for remaining sequences. Since two thousand sentences of length seven are remaining to be processed in the dataset, all the steps have to be repeated for all the sentences which length is equal to seven and the next word will be predicted.

A. Dataset

The IIT Bombay developed this dataset at the Indian Language Technology Center, IIT Bombay for several years. The dataset was collected from a plethora of already provided sources and corpora [12]. This dataset denotes English-Hindi corpus which can be processed for translation. The Hindi text from this corpus is used for building the model. The no of sentences present in the dataset is 15,61,841. The no of sentences used in the project is 2615. These are sentenced having more than six words in a sentence. The sentences which have the length of seven words are taken in the project for further computations. The dataset contains a large number of short sentences with sequences less than of length seven. The words which make up the sequence in the dataset are

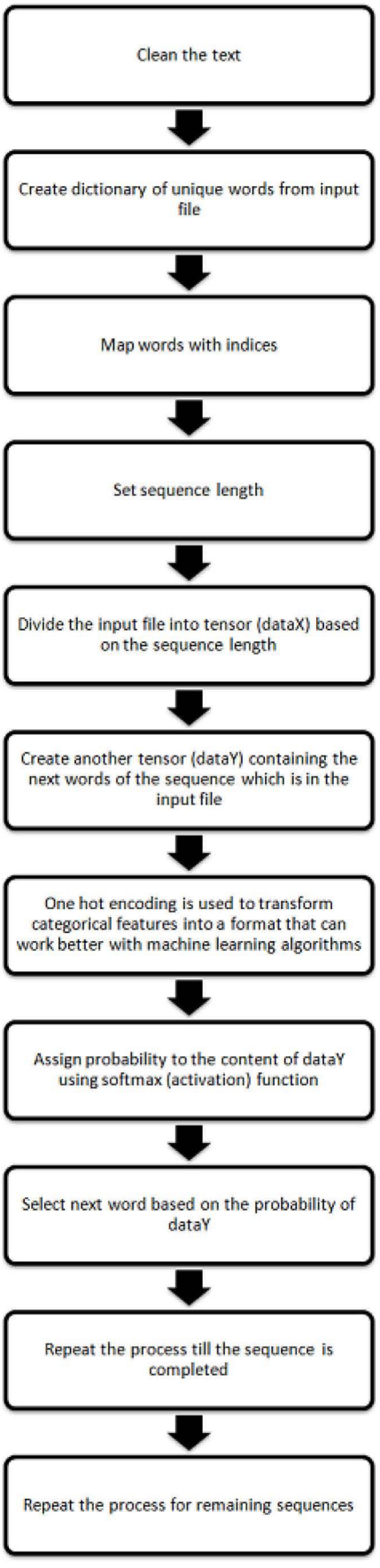

Fig. 1. Flowchart

of Devanagari Hindi script. This dataset is widely used for translation based objective [12]. A validation split of 80:20 is done. The number of sentences used for validation is 523, and that used for training is 2092.

B. LSTM (Long Short Term Memory)

Long-Term Short Term Memory (LSTMs) is a variant of RNNs which are capable of learning Long Term Dependencies, and thus are widely used for Natural Language Generation . LSTMs have memory and remember past data from the input for longer durations of time. They can selectively remember or forget things. They are well suitable for written data inputs, as any word in a sentence is related to words around it (previous and upcoming words) [13].

Each repeating module in LSTM consists of 4 neural network layers that interact with each other. The cell state in LSTMs decides when to read, write and what should be stored in memory. It has gates which conditionally let information pass through them, adding or removing it from cell [14].

Gates work using pointwise multiplication, pointwise addition, and sigmoid function. The output is from 0 to 1 signifying how much information should pass through, where 0 represents no information passes through, and 1 means all information passes through. LSTM has three gates: Forget gate, Input gate, and Output gate.

C. BI-DIRECTIONAL LSTM

Sequence classification problems can be improved by extending the bidirectional LSTMs to enhance model performance. LSTMs are trained twice in bidirectional LSTMs on the input given by the user [15]. The first iteration starts with the forward iteration and passing of sequence like the way LSTMs works and the second iteration is the reverse iteration on the input sequence.

Bi-LSTM processes the data from start-to-end in one iteration and from end-to-start in the other iteration. The prediction is done concerning the future and past of the data as the traversal is in both directions [16]. Bi-LSTMs are preferable over LSTMs because they provide feedback input to the successor layer. This functionality of Bi-LSTMs adds the advantage of complete and faster processing and learning on the input sequence [17].

On comparing the two algorithms, Bi-LSTM is a clear win as it processes the sequence in two iterations, one from front-to-back and other from back-to-front. The use of bidirectional LSTMs provide you with the added advantage of history and near future which makes the prediction accurate and faster. Bidirectional LSTMs (BiLSTM) has different layers which are fed by the learning algorithm to learn long-term dependencies of both the history and future data.

IV. Results

A model has been proposed in this paper to predict the next word in Hindi, given a sequence of at least six words using LSTM and Bi-LSTM Neural Networks. For both the

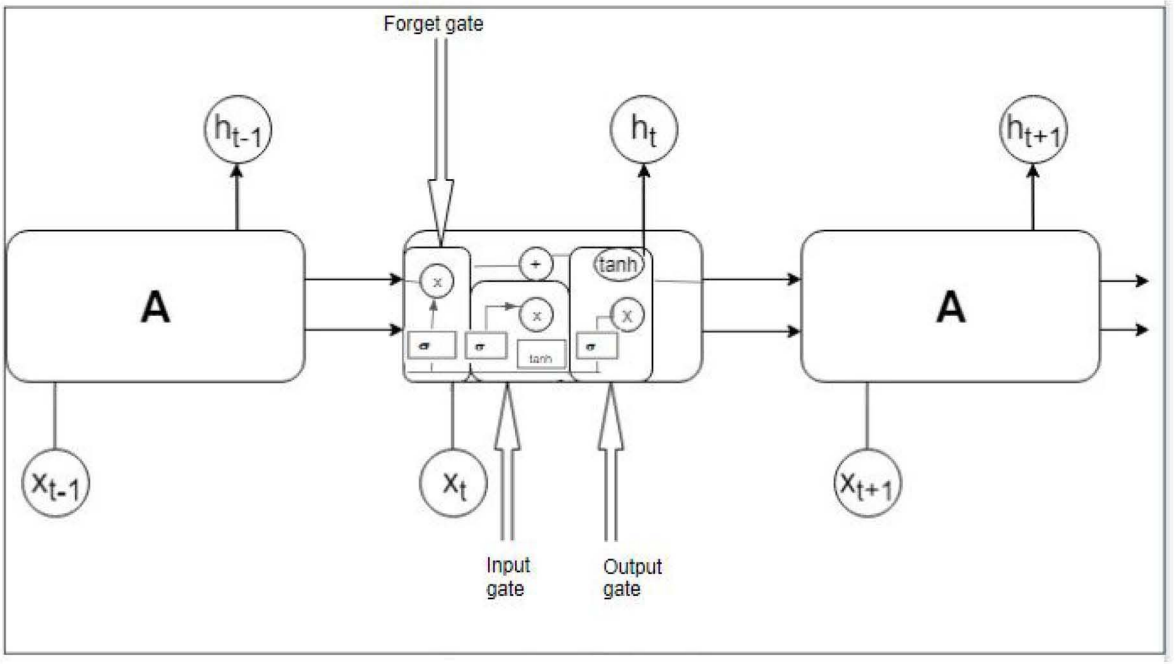

Fig. 2. LSTM Network Architecture

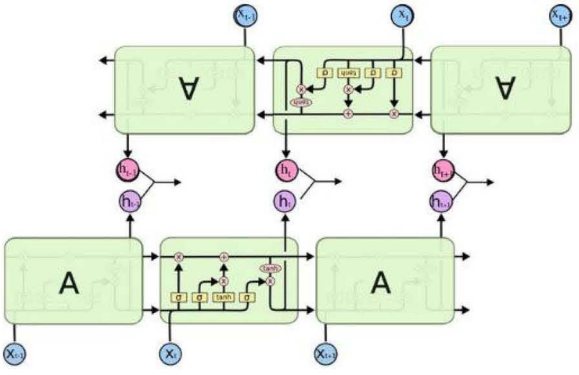

Fig. 3. Bi-LSTM Network Architecture [14]

models softmax activation function and categorical cross entropy loss function was used. Softmax activation function is a mathematical function that is used to obtain the probability that a word will be predicted on the basis of the score for each word obtained from the neural network model thus obtaining the probability distribution of all the words present in the dictionary. The function is defined as

$$\sigma(\mathbf{s})_i = \frac{e^{s_i}}{\sum_{j=1}^{K} e^{s_j}} \tag{1}$$

where $\sigma(\mathbf{s})_i$ is the probability and s_i is score for i^{th} word in the dictionary [18].

Since the task of predicting the next word is a case of multi class classification, Categorical Cross Entropy Loss is used for calculating the loss of model and hence for updating the weights. The function is defined as

$$CE = -\sum_{i}^{C} t_i log(\sigma(\mathbf{s})_i) \tag{2}$$

where CE is Categorical cross entropy loss, t_i denotes the actual truth value and $\sigma(\mathbf{s})_i$ is the probability for the i^{th} word of the dictionary.

A learning rate of 0.001 was used while training both the models. Through cross-validation the accuracy of both the models was calculated by comparing predicted word with actual word in the dataset.

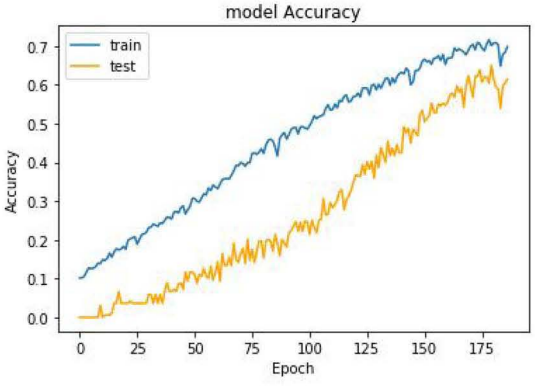

Fig. 4. Accuracy of LSTM Model

Recurrent Neural Networks(RNN) faced a major problem of vanishing gradient [9] due to which front layers of the network might learn slowly. LSTM and Bi-LSTM overcame

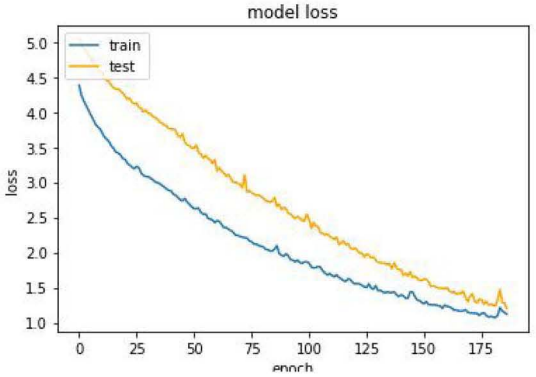

Fig. 5. Loss of LSTM Model

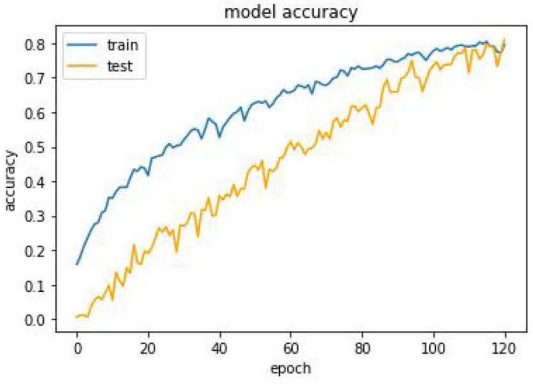

Fig. 6. Accuracy of Bi-LSTM Model

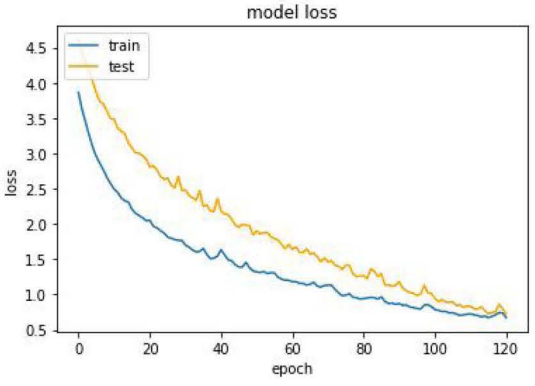

Fig. 7. Loss of Bi-LSTM Model

TABLE I
COMPARISON OF ML TECHNIQUES

	LSTM	*Bi-LSTM*
Number Of Epochs	197	121
Accuracy	0.7089	0.7954
Validation Accuracy	0.5946	0.8107
Loss	1.0730	0.6734
Validation Loss	1.2381	0.7260

this problem. They use gates to selectively forget information. The accuracy of LSTM model obtained is 70.89% and that of Bi-LSTM is 79.54%. However the validation accuracy for both of them is 59.46% and 81.07% respectively. The Bi-LSTM model gives better results as compared to LSTM.
The Bi-LSTM model also learns faster for the same dataset as compared to LSTM due to which the number of epochs was significantly less for the former model.

V. CONCLUSION

A machine learning model to predict the next word for a given sequence of words was built using LSTM and Bi-LSTM models. It was then extended to predict the next few words for the same sequence.
The model developed can be used for predicting the next word in the Hindi Language. This can effectively reduce the number of words that the user has to type, thus increasing the typing speed. It also helps in minimizing the spelling mistakes done by the user. In countries like India, where Hindi is the mother tongue of more than 25% of the population, this system can be a boon.

VI. FUTURE SCOPE

In the future, the system can be extended for other natural language generation tasks like story auto-completion, poem auto-completion, etc.
The model is limited to specific dataset and more randomness can be incorporated in the model by enhancing the scope. The system can be adapted to new words that are not a part of its vocabulary. This adaption will be done when model encounters a new word and adding the word to the vocabulary. This way the model becomes more generalized. The system can be personalized to predict words based on the user's history.

REFERENCES

[1] P. P. Barman and A. Boruah, "A rnn based approach for next word prediction in assamese phonetic transcription," *8th International Conference on Advances in Computing and Communication*, 2018.
[2] R. Perera and P. Nand, "Recent advances in natural language generation: A survey and classification of the empirical literature," *Computing and Informatics*, vol. 36, pp. 1–32, 01 2017.
[3] C. Aliprandi, N. Carmignani, N. Deha, P. Mancarella, and M. Rubino, "Advances in nlp applied to word prediction," *J. Mol. Biol.*, vol. 147, pp. 195–197, 2008.
[4] C. McCormick, *Latent Semantic Analysis (LSA) for Text Classification Tutorial*, 2019 (accessed February 3, 2019). http://mccormickml.com/2016/03/25/lsa-for-text-classification-tutorial/.
[5] Y. Wang, K. Kim, B. Lee, and H. Y. Youn, "Word clustering based on pos feature for efficient twitter sentiment analysis," *Human-centric Computing and Information Sciences*, vol. 8, p. 17, Jun 2018.
[6] N. N. Shah, N. Bhatt, and A. Ganatra, "A unique word prediction system for text entry in hindi," in *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, p. 118, ACM, 2016.
[7] M. K. Sharma and D. Smanta, "Word prediction system for text entry in hindi," *ACM Trans. Asian Lang. Inform. Process*, 06 2014.
[8] R. Devi and M. Dua, "Performance evaluation of different similarity functions and classification methods using web based hindi language question answering system," *Procedia Computer Science*, vol. 92, pp. 520–525, 2016.

[9] 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.
[10] D. Pawade, A. Sakhapara, M. Jain, N. Jain, and K. Gada, "Story scrambler - automatic text generation using word level rnn-lstm," *Modern Education and Computer Science*, 2018.
[11] R. L. Bishop and S. I. Goldberg, *Tensor analysis on manifolds*. Courier Corporation, 2012.
[12] A. Kunchukuttan, P. Mehta, and P. Bhattacharyya, "The iit bombay english-hindi parallel corpus," *Language Resources and Evaluation Conference*, 2018.
[13] M. Sundermeyer, R. Schlüter, and H. Ney, "Lstm neural networks for language modeling," in *Thirteenth annual conference of the international speech communication association*, 2012.
[14] C. Olah, *Understanding LSTM Networks*, 2019 (accessed March 7, 2019). http://colah.github.io/posts/2015-08-Understanding-LSTMs/.
[15] J. Brownlee, *How to Develop a Bidirectional LSTM For Sequence Classification*, 2019 (accessed April 10, 2019). https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/.
[16] Y. Huang, Y. Jiang, T. Hasan, Q. Jiang, and C. Li, "A topic bilstm model for sentiment classification," in *Proceedings of the 2nd International Conference on Innovation in Artificial Intelligence*, pp. 143–147, ACM, 2018.
[17] S. Stymne, S. Loáiciga, and F. Cap, "A bilstm-based system for cross-lingual pronoun prediction," in *Proceedings of the Third Workshop on Discourse in Machine Translation*, pp. 47–53, 2017.
[18] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.