FaGoN: Fake News Detection model using Grammatic Transformation on Neural Network

Youngkyung Seo
Department of Electrical Engineering
Korea University
Seoul, Republic of Korea
ygseo@korea.ac.kr

Chang-Sung Jeong
Department of Electrical Engineering
Korea University
Seoul, Republic of Korea
csjeong@korea.ac.kr

Abstract— **These days, most of fake news are detected and verified by people, which requires a great amount of time and effort. It is difficult to figure out the truthfulness of the news by machine algorithm because the sentences have various forms. In this paper, we shall present a fast and efficient fake news detection model which can figure out whether the given proposition is true or not from article by exploiting grammatical transformation based on deep learning. Our model consists of four layers: word embedding layer, context generation layer, matching layer and inference layer. In word embedding layer, the words in proposition are embedded into word vector. In context generation layer, the word vectors enter into LSTM layer and generate context vector. In matching layer, attention vector is generated from the contextual embedding vector in the previous layer computing the weighted sum. Then, the hidden state vector from LSTM layers and attention vector are compared through matching operation generating the sentences which has the same meaning but different forms. In inference layer, our model calculates the similarity between the generated sentences and the sentences in articles, and classifies the answer, true or false. We shall evaluate our model calculating the perplexity to figure out whether the generated sentences are grammatically correct. Also, the model is tested by changing the sentence group's size to find the optimal size of the group. By showing the Our model figured out the fake news very well with the test of CNN news dataset getting the right answer.**

Keywords— Natural Language Processing, Fake news, Sequence to sequence, Neural network, Deep learning;

I. Introduction

The amount of fake news is increasing with various purposes such as politic issues to create public sentiment by distributing false information. There are many attempts for development of fake news detection system these days[1]. However, mostly the truth of the news is verified by people in those systems as it's difficult to find truthfulness with machine algorithm when the sentences are transformed. In order to spot fake news without people, it is important to use the model using neural network to generate various forms of the sentences with the same meaning and calculating the similarities of the proposition and the article.

In this paper, we shall show a new model which can figure out the truthfulness of the articles by comparing the given proposition with the articles. Our model consists of four layers: word embedding layer, context generation layer, matching layer and inference layer. In word embedding layer, the words in proposition are embedded into word vector. In context generation layer, the word vectors enter into LSTM layer and generate context vector. In matching layer, attention vector is generated from the contextual embedding vector in the previous layer computing the weighted sum. Then, the hidden state vector from LSTM layers and attention vector are compared through matching operation generating the sentences which has the same meaning but different forms. In inference layer, our model calculates the similarity between the generated sentences and the sentences in articles, and classifies the answer, true or false. We shall evaluate our model calculating the perplexity to figure out whether the generated sentences are grammatically correct. Also, the model is tested by changing the sentence group's size to find the optimal size of the group. By showing our model figured out the fake news very well with the test of modified CNN news dataset from DeepMind getting the right answer.

The rest of our paper is as follows: In section 2, we describe related works, and in section 3, present our model for fake news detection based on neural network with two parts, sentence generator part and inference part. In section 4, we explain the experimental results, and in section 5, give a conclusion.

II. Related Works

A. Deep Neural Network for language model

In natural language processing, RNN(Recurrent Neural Network) is used for the language model to process the sequence between characters and words[2]. RNN is one of the neural networks that the hidden nodes are connected with the directional edge. Every hidden node of the sequence uses the same weight, and the output is affected by the previous computation output since the neural network is recurrent. Therefore, RNN is suitable to process the sequential data such as text and sound. It is a flexible structure that can be used in various kinds of dataset regardless of the length of its input and output. However, when the length of the input is getting large, the gradient decreases which leads to bad performance of training. LSTM(Long Short Term Memory) is developed to solve this vanishing gradient problem[3].

LSTM unit consists of various gates, and it has the structure of RNN which has additional cell states from the gates. The cell state generated from computing forget gate, input gate and output gate effects the next cell that the gradient can be delivered to the longer cells. Forget gate computes the hidden states to decide whether the previous hidden state will be effective to the present hidden state. Input gate decides whether the next hidden state uses the hidden state of input. The output gate computes the final output state with the cell state from forget gate and input gate.

For longer delivery of the cell state, Bi-LSTM(Bidirectional Long Short Term Memory) is developed[4]. It has two hidden layers: the hidden layer which has the forward states information and another hidden layer which has the backward states information. The Bi-Directional LSTM improves the performance considering the hidden state forward and backward.

B. *Sequence to sequence learning*

Sequence to sequence learning is a model to predict the sentence using RNN, LSTM, GRU, or Bi-LSTM for implementation[5]. This model can process a complex and large amount of sequence data using deep layer of RNN. Furthermore, it is useful to use the unfixed size of input and output such as text and sound. It has the input of source sequence and target sequence, or parallel corpus for training. The input is processed with two parts: encoder and decoder. The encoder translates the source sequence into fixed size vector which is the size of vocabulary, and the decoder translates the vectors of encoder into target sequence. When the model predicts the target sequence, decoder uses the previous predicted word as an input to predict the next word.

Attention mechanism is a powerful method to predict the target sequence in sequence to sequence learning[6,7]. In attention layer, inner vector of decoder's hidden state and encoder's hidden state is computed as attention vector. The attention vector is changed into the probability using softmax function. The context vector is computed with the weighted sum of the hidden state of decoder and the attention vector from encoder. The hidden state of decoder and the context vector are combined to decode maintaining the memory of the previous encoder's hidden state.

Sequence to sequence learning is mostly used in language model for translation system and chatbot system since the amount of grammar rules are large, and there are so many exceptions. This model is used in Google neural machine translation system(GNMT) with attention mechanism[8].

C. *Beam search decoder*

The sequence to sequence learning needs a decoder to decode the vectors into sentences which consist of the word with the highest possibility from the softmax layer. There are various search methods to decode the words in sequence to sequence learning.

Greedy search decoder decodes one word which has the highest probability in each step. The calculation is easy and simple but it doesn't consider the probability of the entire sentence. So it's hard to get the optimal solution from greedy search decoder[9].

Beam search decoder considers the given number of high probability words in each step, and selects the combination which has the highest probability. Therefore, it can produce more candidates with the beam size of candidates. We use beam search decoder to have multiple candidates for generating several output sentences which have the same meaning but different forms[10].

III. MODEL

We present our model in detail in this section. It consists of four layers: word representation layer, context representation layer, matching representation layer and decision layer as shown in Fig.1. After generating sentences, the similarity between the sentences from our model and the article are compared respectively for getting the final answer.

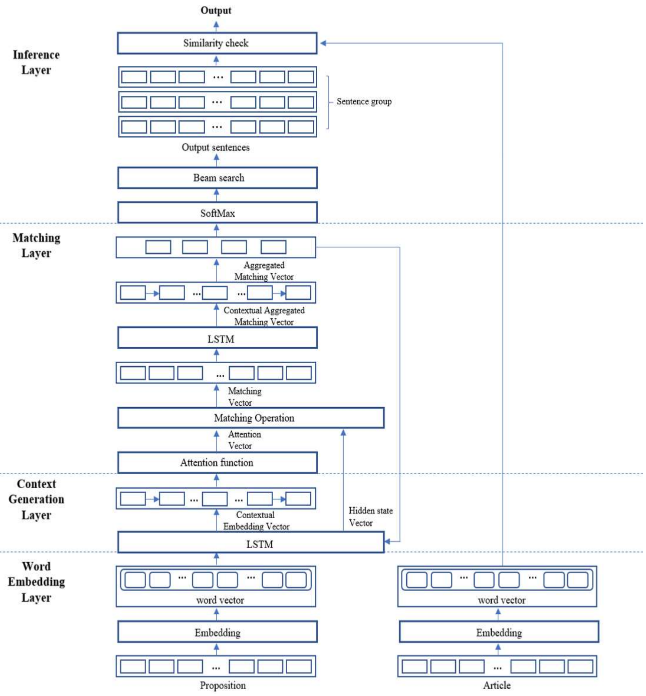

Fig. 1. The overall architecture of our model

- Word embedding layer: Each word from the proposition is translated into vector with one-hot encoding. It generates the word vector through the embedding.
- Context generation layer: This layer outputs a contextual embedding vector for the word vector through LSTM. In this layer and the matching layer, we make use of sequence to sequence learning for training and generating the sentences. We use two layers of LSTM for higher accuracy. The contextual embedding vector is generated by computing a weighted sum of the word vector.
- Matching layer: This layer outputs the matching vector through matching operation. We use attentive matching operation. Then, the matching vectors are generated from the two layers of LSTM. The output for aggregated matching vector is concatenated from the last matching vectors in LSTM models. The aggregated matching vector is used in context representation layer.

- Inference layer: This layer determines the probability between the words by using aggregated vectors through the networks and then makes the given number of words with softmax function. The beam search combines the sequence with the word combinations which have the high possibilities. The sentences generated from our model and the article are embedded with Doc2Vec model[11]. We use the pretrained Doc2Vec model for embedding. The cosine similarity between vectors from each sentence in generated sentence group and article is computed to compare the similarity. The maximum similarity value of the group is used for getting the answer true or false. When the similarity is more than 0.5 at the range of 0 to 1, the proposition is classified as True.

IV. Experiment

We tested the dataset and adjusted the model in python 3.0.1, Pycharm environment with 1 CPU(Intel® Core® CPU @ 3.50GHz) and memory of 8GB as well as GPU by NVIDIA for the experiments. Tensorflow library is used to set the deep learning model and other parameters.

A. Experiment Dataset

We tested our model with two types of dataset. First, the parallel corpus dataset is used as an input for generating the sentences when sequence to sequence learning model is exploited. Second, we modified CNN news articles from DeepMind and created various types of propositions to evaluate the performance of inference[12].

The dataset used for the training when the model generates the sentences in context generation layer and matching layer is SNLI(The Stanford Natural language Inference) corpus provided by Stanford university[13]. The parallel corpus dataset consists of the set of a sentence and another sentence which is a grammatically transformed or includes the synonyms. We filtered the training dataset with the maximum length as 30 for the efficient training.

To evaluate the inference, we used CNN news dataset to compare the sentence group generated from the layers to the articles about the proposition with the article. The original CNN news dataset consists of questions and news articles with empty entities. We fill the empty entities generating the complete article for the test and use the question as the proposition by creating 4 types of propositions: using synonym or acronym, adding the words, modifying the word and modifying passive or active form. The following Table 1 and 2 show the specifications of each dataset that are used to evaluate the model.

TABLE I. THE CORPUS DATASET ATTRIBUTES

Attribute type	Size
Number of corpus dataset	4,540
Number of epoch	19,425
Number of vocabulary	18,216
Maximum length of the sentences	30

TABLE II. THE CORPUS DATASET ATTRIBUTES

Attribute type	Size
Number of synonym or acronym questions	50
Number of word addition questions	50
Number of word modification questions	50
Number of passive or active form	50
The total number of proposition dataset	200
The total number of article	200

B. Experiment Result

In this section, we describe the experiments of our model's accuracy. First, we evaluate the generated sentences from sequence to sequence model with perplexity. Second, we test the model with accuracy changing the generated sentence group's size by controlling the beam size. Third, we have the experiment with our model whether the fake news is detected very well with four types of propositions.

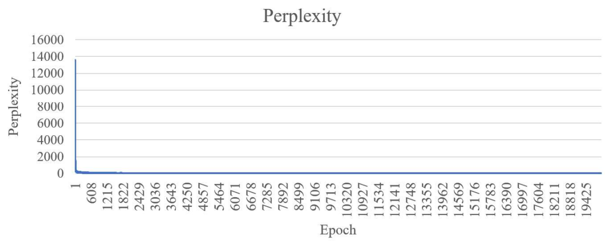

Fig. 2. Perplexity of our model by increasing the time step

First, we evaluate the generated sentences from sequence to sequence model with perplexity. The perplexity is calculated to figure out whether the generated sentences are grammatically correct and complete. It is a measurement of how well our model predicts the next word in a sequence by calculating the probability of the words. A low perplexity indicates the probability is good at predicting the sample that is the generated sentence is complete. When there are m number of sentences such as s_1, s_2, …, s_m, we can present the probability under our model as the following equation.

$$\prod_{i=1}^{m} p(s_i) \tag{1}$$

The equation (1) can be represented with the log probability as equation (2). Using (2), perplexity is represented by calculating exponent for figuring out the distribution more clearly when M is the total number of words in the test data.

$$\log \prod_{i=1}^{m} p(s_i) = \sum_{i=1}^{m} \log p(s_i) \tag{2}$$

$$\text{Perplexity} = 2^{l}; \text{where } l = \frac{1}{M}\log p(s_i) \qquad (3)$$

As the training starts about 20,000 number of epochs, the perplexity decreases as about 1.2 which shows that the model predicts the next words very well in Fig. 2.

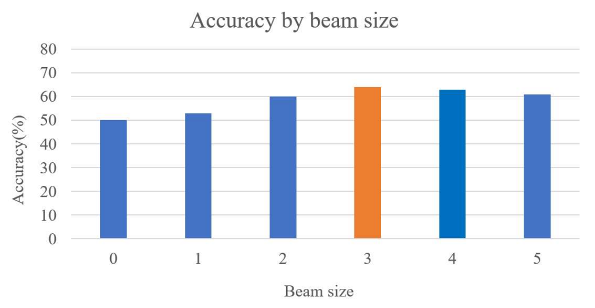

Fig. 3. Accuracy of our model by changing the beam size

Second, we find the optimal sentence group's size for the highest accuracy with our model by changing the beam size as 0 to 5 as shown in Fig.3. When the beam size is 0, we evaluate the model without generating sentences using only proposition when inferencing the answer. We get the answer by computing cosine similarity between the sentences in group and article. The accuracy is the lowest when only the proposition is used for getting the answer. However, using the suitable amount of generated sentences increases the accuracy. We get the highest accuracy when the beam size is set as 3 which uses 3 sentences for the inference. Our inference part shows better accuracy about 10% than the model with the proposition.

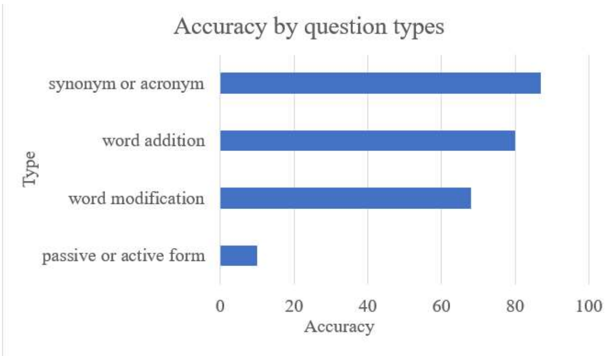

Fig. 4. Accuracy of our model by the types of propositions

Third, we make use of the various types of propositions to our model. There are four types of propositions using synonym or acronym, adding words, modifying words and changing the form as passive or active form. Our model catches out the fake news accurately with the synonym or acronym propositions. Fig. 4 shows that our model figures out the truthfulness of the news relatively better in modification of words than the form is changed as passive or active form.

V. Conclusion

In this paper, we have shown a new model which can detect the fake news with high accuracy exploiting the given proposition. Our model consists of four layers: word representation layer, context representation layer, matching representation layer and decision layer.

We have tested our model with two types of dataset. First, the parallel corpus dataset is used as an input for generating the sentences when sequence to sequence learning model is exploited. Second, we modified CNN news articles from DeepMind and created various types of propositions to evaluate the performance of inference.

For the experiment, we evaluate our model by calculating the perplexity of the generated sentences. Furthermore, we change the generated sentence group's size by controlling the beam size and find the optimal size. Also, we have the experiment with our model whether the fake news is detected very well with four types of propositions.

As a future work, we shall make use of our model to other languages with larger dataset. We shall continue our work with implementing it on distributed parallel environment for fast training and processing of fake news detection.

Acknowledgment

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2017R1D1A1B03035461), the Brain Korea 21 Plus Project in 2018, and the Institute for Information & communications Technology Promotion(IITP) grant funded by the Korean government (MSIP) (No. 2018-0- 00739, Deep learning-based natural language contents evaluation technology for detecting fake news).

References

[1] M. Gahirwal, S. Moghe, T. Kulkarni, D. Khakhar and J. Bhatia, "Fake News Detection", International Journal of Advance Research and Innovations in Technology, vol. 4, no. 1, pp. 817-819, 2018.

[2] T. Mikolov, M. Karaflat, M. Burget, L. Cernocky and S. Khudanpur, "Recurrent Neural Network based on Language Model", Eleventh Annual Conference of the International Speech Communication Association, 2010.

[3] M. Sundermeyer, R. Schluter and H. Ney, "LSTM neural networks for language modeling", Thirteenth Annual Conference of the International Speech Communication Association, 2012.

[4] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures", Neural Networks, vol. 18, no. 5-6, pp. 602-610, 2005.

[5] I. Sutskever, "Sequence to Sequence Learning with Neural Networks", Advances in Neural Information Processing System, pp. 3104-3112, 2014.

[6] M.T Luong, H. Pham and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation", arXiv preprint arXiv: 1508.04025, 2015.

[7] D. Bahdanau, C. Kyunghyun and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate", arXiv preprint arXiv:1409.0473, 2014.

[8] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, and J. Klingner, "Google's Neural Machine Translation System: Bridging the

gap between human and machine translation", arXiv preprint arXiv:1609.08144, 2016.

[9] P. Langlais, A. Patry and F. Gotti, "A Greedy Decoder for Phrase-Based Statistical Machine Translation", Conference of Theoretical and Methodological Issues in Machine Translation, 2017.

[10] C. Hokamp and Q. Liu, "Lexically Constrained Decoding for Sequence Generation Using Grid Beam Search", arXiv preprint arXiv:1704.07138, 2017.

[11] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents", International Conference on Machine Learning, pp. 1188-1196, 2014.

[12] K. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman and P. Blunsom, "Teaching Machines to Read and Comprehend", arXiv preprint arXiv:1506.03340v3, 2015.

[13] P. Young, A. Lai, M. Hodosh and J. Hockenmaier, "From Image Descriptions to Visual Denotations: New Similarity Metrics for Semantic Inference Over Event Descriptions", Transactions of the Association for Computational Linguistics 2, pp. 67078, 2014.