A Deep Learning Approach for Text Generation

Ahmed Elmogy[1], Belal Mahmoud[2], and Mohamed Saleh[2]
[1] Computer Engineering Dept., Prince Sattam Bin Abdelaziz Univ., KSA, and
Computers & Control Eng. Dept, Tanta, Egypt (email:a.elmogy@psau.edu.sa)
[2] Computer Science Dept., ISSR, Cairo Univ., Egypt
(email:eng_belalsaeed@pg.cu.edu.eg, mohamed_ezzeldin@pg.cu.edu.eg)

2019 29th International Conference on Computer Theory and Applications (ICCTA) | 978-1-7281-5276-9/20/$31.00 ©2019 IEEE | DOI: 10.1109/ICCTA48790.2019.9478833

Abstract— **One of the most challenging language modeling problems is text generation. The importance of language modeling comes from its involvement in many language processing tasks such as conversational system, speech to text, and text summarization. The language models are typically trained to learn the occurrence of next word in a sequence based on previous words in the text. However, when it comes to testing, it is highly expected that the entire sequence will be generated from the scratch which is computationally not suitable to many applications. In this paper, one of the popular deep learning architectures called bidirectional recurrent neural network (BRNN) to develop a text generation approach. The recurrent neural network used in the developed approach uses long short-term memory (LSTM). By using LMST recurrent neural network, the proposed approach is well suited to making predictions based on time series data. Two representations are considered in this paper; word to vector (word2vec) and one-hot representations. The word2vec model is used with two datasets called Twilight and Alice in Wonderland stories while the one-hot model is used only with Alice in Wonderland story only. The experimental results show that the word2vec model outperforms the one-hot model when using to large data sets.**

Index Terms— **Text generation, Deep learning, BRNN, LSTM, Word2vec, one-hot vector**

I. INTRODUCTION

NATURAL language processing (NLP) is the application of computational techniques to the analysis and synthesis of natural language and speech [1]. Text generation is the branch of NLP that enable users to draw insights, create advertisements, help you text and more. Text generation can be found in many applications that include but not restricted to machine translation [2], video/text summarization [3], and story generation [4]. From the perspective of machine learning, text generation is concerned with predicting the correct sequence of words given some context. Language models based on n-grams [5], feed-forward neural networks [6], and RNN [7] are the most popular models used for the implementation of text generation.

RNN is one of the most efficient models that have been used for text generation. The power of using RNN in text generation comes from its ability to memorize its internal state to process sequences of inputs. Bidirectional recurrent neural networks (BRNN) [8] are like RNN but with ability to learn from both sides. Using such networks in text generations helps in learning text meanings and relation between words. RNN are often called Long short-term memory (LSTM) [9] networks as LSTM is the building unit for its layers. RNN composed of LSTM units is often called an LSTM network [10]. The LSTM unit consists of a cell, an input gate, an output gate and a forget gate. The cell is responsible for remembering values over time intervals.

On the other hand, one-hot vector [11] and word2vec [12] are popularly used for the implementation of text generation. One-hot vector is a method for transforming each word in the dictionary to a vector of dictionary length of 0s except the word index in the dictionary is 1. Word2vec method works by transforming the word to several hundred dimensions with each unique word in the corpus being assigned a corresponding vector in the space. These dimensions are presenting the features of each word. In this paper, we propose a Bidirectional RNN with the help of LSTM method to generate new story based on model's learning from the data set. Two word embedding methods are implemented; namely word2vector method, and one hot vector. Additionally, we implement the model and compare the performance of the two word embedding methods.

The rest of this paper is organized as follows; Section 2 discusses some related work. Section 3 describes the proposed approach. Section 4 discusses the results. Finally, Section 5 concludes the paper.

II. RELATED WORK

Natural language is the most natural form of communication for humans. It is therefore essential that interactive AI systems can generate text. Natural language processing (NLP) is one of the most popular AI areas that is concerned with how to program computers efficiently in order to be able to analyze and understand human languages. The main power of NLP is its ability to work on huge amounts of language data. Challenges of tackling the NLP field include but not restricted to natural language understanding, and natural language generation. The language generation is the main concern of this paper. Many models and architectures have been proposed in the literature to tackle the text generation problem such as n-grams [5], feed forward neural networks [6], RNN [7]. RNN is

known as one of the popular deep learning architectures that has been used over years in many applications. Deep learning works by extracting higher level features from raw data through multiple layers learning. Thus, by using deep learning algorithms and architectures, a hierarchy of features are extracted which increases the efficiency of the semantic understanding of low level data.

RNN [7] employed in this paper works on the principle of looping the information in every layer of the network and thus able to save the present output and use it to predict the next output of the network. From the context of text generation, RNNs are able to connect previous information to the present task. For example, consider we are building a model trying to predict the last word in sentence "Cairo is the capital of ……" based on the existing previous words. It is very obvious that the last word is Egypt. RNNs are very efficient in modeling such cases because of their ability to deeply learn to the past information. However, RNNs are not with no restrictions. The main restriction of RNNs is that, the future input information cannot be reached from the current state. BNRRs are based on the idea of connecting two opposite directions hidden layers to the same output. Thus, by using BRNNs, future input information is reachable from the current state. Even with great power of BNRRs, they are not suitable to be used the gap between some relevant information and the place it is needed is large. This is called long dependency problem [8,10]. LSTM network is another kind of RNNs which is mainly developed to avoid the long dependency problem. The default behavior of the LSMT network is remembering information for long periods of time.

In this paper, a text generation approach that is based on LSTM method is developed to generate text. Two text embedding methods are tested in the proposed model; one-hot method and word2vec method. One-hot vector method is used to map each word in the text to one single word in the vocabulary. Word2vec method maps each unique word in the corpus to its corresponding vector in space.

III. PROPOSED MODEL

This section explains the details of the proposed approach shown in figure 1. The proposed approach can be described as follows**:**

1. **Data preprocessing**: this is done by:
   - Transforming all letters to small form to learn the model that the same words are equal does not matter if the first letter is capital or small. For example: Sun is equal to sun.
   - Replacing short form words with its original form. For example replacing "I'm" with "I am". One of the problems that needs more investigations is how to employ "'s" form as it has different meanings depending on the phrase. For example: John's book, and John's good, "'s "has different meanings in the two sentences.
2. **Converting the corpus into list of words:** this is done by removing unnecessary characters, splitting the corpus to words only with no commas, no hyphens, and no full stops.
3. **Create the dictionary**: We created the dictionary of the data set which contains the distinct words in the corpus.
4. **Use word2vec & one-hot models**: in this step, our approach is concerned with transforming the words into a form that model can understand. Two models have been developed in the proposed approach; each one is using a different method for text generation. The first model uses one-hot vector method which transforms each word in the dictionary to a vector of dictionary length of 0s except the word index in the dictionary takes 1. The second model uses word2vec method which transforms the word to several hundred dimensions (three hundreds in this model) with each unique word in the corpus being assigned a corresponding vector in the space. these dimensions are presenting the features of each word.
5. **Create sequence of words**: We created sequences of words with 30 words size each, which will be used in learning by learning the next word prediction according to the previous 30 words.
6. **Create the BRNN**: we are using Keras [15] for this purpose as it provides a very good abstraction to design neural networks architectures. the following neural network is employed in our approach:
   a) Embedding layer: This layer can only be used as the first layer in a model. it turns positive integers (indexes) into dense vectors of fixed size. In our model, its input is vocabulary size and its output is the number of dimensions in word to vector, which is 300. This layer is used only in word2vector model.

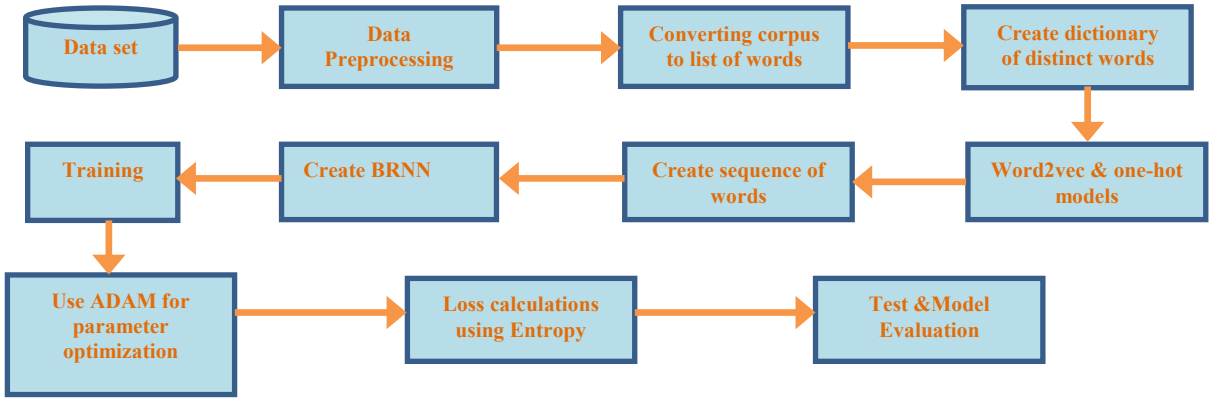

Fig.1 Proposed approach

b) Bidirectional LSTM layer: with size of 256 and using RELU as activation.
c) Dense layer of the size of the vocabulary, a dense layer is simply a layer where each unit or neuron is connected to each neuron in the next layer.
d) Finally, a softmax activation layer

7. **Using ADAM [16] optimization method** to optimize the parameters of the built BNRR. The loss calculation is done on the categorical cross-entropy [17]. The network should provide us with the probability for each word from the vocabulary to be the next one after a given sentence.

IV. EXPERIMENTAL RESULTS

The datasets used to test and evaluate our approach are Alice in Wonderland [18] and Twilight [19] stories. Alice in Wonderland story contains 30318 words and vocabulary with 2984 words. This is considered as a small corpus. However, this data set is chosen because of its data size constraint which makes it suitable to be used on colab website. Twilight data set Contains 630529 words and vocabulary with 15626 words, and we will be used only with word to vector method.

Table 1 shows the results of our proposed model with two embedding methods; word2vec and one-hot vector. The performance of the proposed model is assessed using the accuracy and loss function metrics. As shown, the Word2vec model with Wonderland dataset has the best results according to the accuracy and loss function. However, it has the worst meaningful output because of the small amount of data in the data set so the model cannot learn the similarity between words. For example, the model finds out that the most similar words to the word " boy " are (lake, stolen, image, yawing, melancholy, met) with similarity between (.16 to .18) which is very bad as shown in figure 3. The T-sne method is used for the representation of words**.**

TABLE 1 COMPARISON BETWEEN RESULTS

Metrics	Models type Text generation with word2vec (Wonderland dataset)	Text generation with word2vec (Twilight dataset)	Text generation with one hot vector
Start loss Fn	7.4388	8.7588	6.7612
End loss Fn	0.0079	0.02716	0.13869
Start accuracy	0.0331	0.03541	0.0348
End accuracy	0.9989	0.9812	0.9608
Epochs	120	80	160

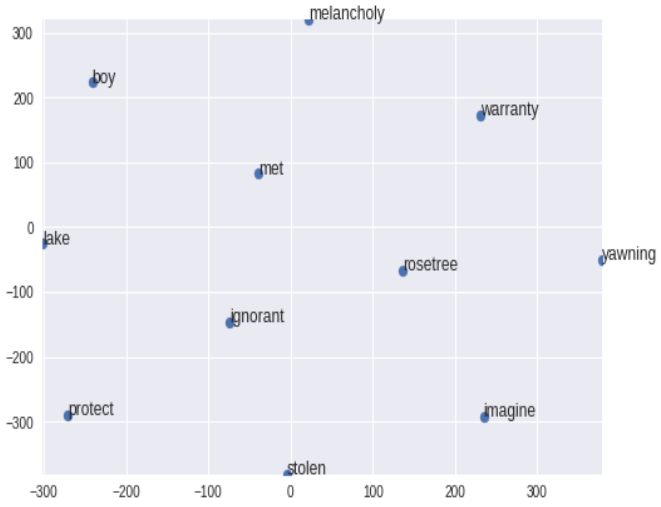

Fig. 2 t-sne representation of word "boy"

The test input used to assess the performance of the proposed model is "she took down a jar from one of the shelves as she passed". This input is used with both word2vec and one-hot vector models. The output when using word2vector is: "yelp australia unfortunate ix butterfly today facility is note howling fireplace ornamented brandy puss uglification considerable royalty flappers wonderland breach give blew magpie poured checked despite be teacups walks repeating bowed uncommonly character bells head answers leaving eating liked org subjects hurrying treated refund matter conger memory couples site row". The output for one-hot vector is " and is is the lives she was close and as off when she saw the whole little little sharp key so she saw the back of the whole pack at crowded at one round she had read out at the cook of time when the baby took out it". Figures 3 & 4 show the loss functions of both models and figures 5 & 6 show the accuracies.

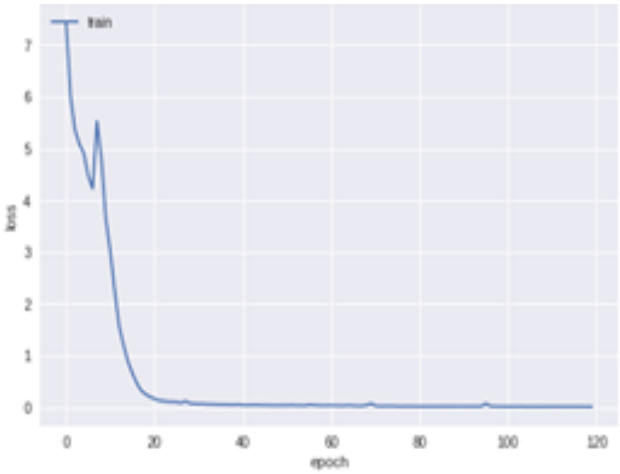

Fig. 3 word2vec model loss function

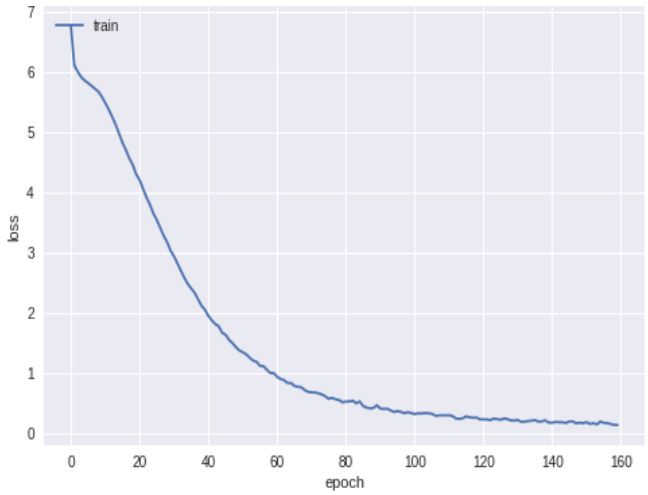

Fig. 4 one-hot vector model loss function

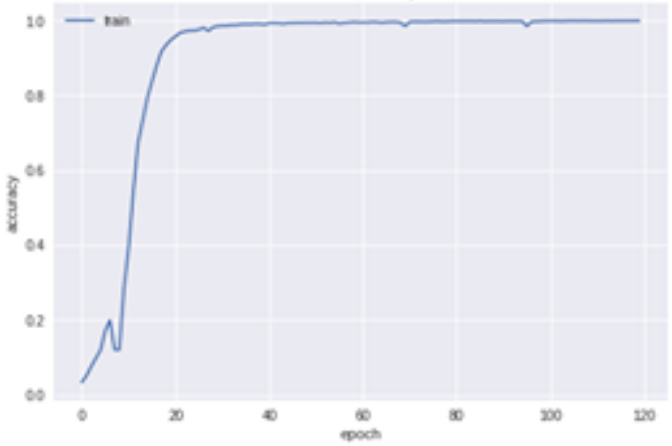

Fig. 5 word2vec model accuracy

As seen the figures, the one-hot vector gets worse results than word2vec in terms of loss function and accuracy. But, more meaningful results that word2vec are obtained. It treats each word in the corpus as a unique word.

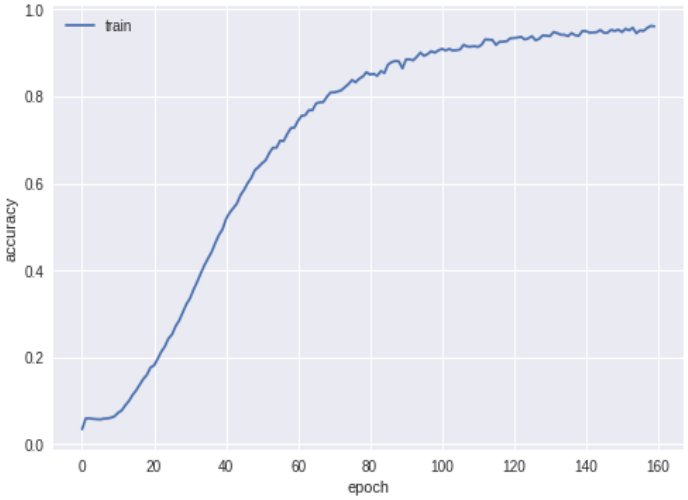

Fig. 6 word2vec model accuracy

Also, because of the small amount of dataset in this story, better results are obtained in the output. When using word2vec model with Twilight story, we get more meaningful results than the outputs for both models with wonderland story because of the large amount of corpus in Twilight story.

V. CONCLUSION

NLP is the most effective way to allow computers to communicate with humans. In our approach, we used language models to generate stories. These models are typically trained to predict the next word in a sequence given the previous words. However, at test time, the model is expected to generate the entire sequence from. A corpus of Wonderland and Twilight stories as used data sets in this paper. Based on the corpus, we used bidirectional recurrent neural network (BRNN) for the text generation with the help of long short-term memory (LSTM). Two embedding methods; one-hot vector, and word2vec are used to test and evaluate the proposed approach. The proposed model is able to generate some text based on the Wonderland and Twilight stories. The results show that using word2vec model with Wonderland story has the best performance in terms of accuracy and loss function. However, the results are unreadable with Wonderland story due to the small size of the data set so it tends to over-fit. The results with Twilight dataset is much better because of the large amount of the data in the data set. The one-hot model generates a better text for Wonderland story as it treats every word as unique word so no fake similarity comes from the small dataset.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of the Deanship of Scientific Research at Prince Sattam Bin Abdulaziz University under the research project # 2019/01/11117.

REFERENCES

[1] W. Ogallo W and A. Kanter, " Using Natural Language Processing and Network Analysis to Develop a Conceptual Framework for Medication Therapy Management Research", AMIA Annual Symposium proceedings, 2017.

[2] D. M. Matthias, W. Axel-cyrille and n. Ngomob,"Machine Translation using Semantic Web Technologies: A Survey", journal of web semantics, vol. 51, pp. 1-19, 2018.

[3] M. A. Fattah and F. Ren, "GA, MR, FFNN, PNN and GMM based models for automatic text summarization", Computer Speech & Language, Vol. 23, No. 1, pp.126-144, 2009.

[4] F. Angela M. Lewis and Y. Dauphin, "Hierarchical Neural Story Generation", , Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pp. 889–898, 2018.

[5] Sidorov Grigori , Castillo Francisco, Stamatatos Efstathios , Gelbukh Alexander , and Chanona-Hernández Liliana," Syntactic N-grams as machine learning features for natural language processing", Expert Systems with Applications: An International Journal. Volume 41, No. 3, pp. 853-860, 2014.

[6] P. Tahmasebi and A. Hezarhani," Application of a Modular Feedforward Neural Network for Grade Estimation. Neural Resources Research, 2011.

[7] M. Ranzato, " Sequence level training with recurrent neural networks", In International Conference on Learning Representations, 2016.

[8] Y. Du, W. Wang, and L. Wang," Hierarchical recurrent neural network for skeleton based action recognition", In CVPR, 2015.
[9] M, Sundermeyer," LSTM neural networks for language modeling". In INTERSPEECH, 2012.
[10] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki," Scene labeling with LSMT recurrent neural networks", In CVPR, 2015.
[11] D. Harris and S. Harris," Digital Design and Computer Architecture", Morgan Kaufmann Publishers, Massachusetts, 2007.
[12] T. Mikolov, " Distributed Representations of Words and Phrases and their Compositionality", In Advances on Neural Information Processing Systems, 2013.
[13] P. Jain, "Story Generation from Sequence of Independent Short Descriptions" In Proceeding of Conference on Knowledge Discovery and Data Mining, 2017
[14] Jaya, A., and Uma, G.V.," An intelligent system for semi-automatic story generation for kids using ontology. Proceedings of the third Annual Computer Conference, Bangalore Conference, 2010.
[15] Jogo Moolayil, "Learn Keras for Deep Neural Network:A Fast-Track Approach to Modern Deep Learning with Python", Apress, 2018.
[16] Diederik P. Kingma*, Jimmy Lei Ba," Adam: A Method For Stochastic Optimization", in the proceedings of ICLR conference, pp. 1-15, 2015.
[17] X. Li, E. Gavves, C. G. M. Snoek, M. Worring, A. W. M. Smeulders, "Personalizing Automated Image Annotation Using Cross-Entropy", Proc. ACM Multimedia, pp. 233-242, 2011.
[18] Carroll, L., (1865), Alice in Wonderland. Macmillan, London.
[19] Meyer, S., (2005), Twilight series. Little, Brown and Company, Columbus, Georgia.