Yet Another Text-to-Text Generation Model using Character-based Recurrent Neural Network (RNN)

Haris Ijaz[1], Seemab Latif[2]
[1]Dept. of Electrical Engineering,
[2]Dept. of Computing
School of Electrical Engineering and Computer Science
National University of Science and Technology. Islamabad, Pakistan.
[1]hijaz.msee19seecs@seecs.edu.pk
[2]seemab.latif@seecs.edu.pk

Abstract—**Text generation is a basic and very important task of Natural Language Processing (NLP). It is used in variety of NLP application like machine translation, text summarization and in conversational chatbots. But there are many problems in generating natural sentences. Recently there are many developments in recurrent neural network (RNN) especially Long-Short Term Memory is widely used in such scenarios where text generation is required. This paper will provide LSTM-based character level language model implemented using wikipedia dataset to generate text. Adam optimizer algorithm is used to optimize the model. BERTScore and BLEU is used to evaluate the model. Results shows the models outperforms the state of the art models with same settings, including logical models and statistical models.**

Keywords— text generation, RNN , LSTM,

I. Introduction

Commonly for the Natural Language Generation (NLG) can be done through many different approaches to generate text. One is Encoder-Decoder framework[1] and seq-to-seq models[2]. The encoder-decoder method only generates the target sentence from the semantics of the input sequence. other Recurrent Neural Network (RNN)[3] is also used to generate text from input sequences. Recurrent Neural Networks(RNN) includes Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). Recurrent Neural Networks (RNNs) form an expressive model family for sequence tasks. They are powerful because they have a high-dimensional hidden state with nonlinear dynamics that enable them to remember and process past information[4]. Long Short Term Memory (LSTM) [5] is type Recurrent Neural Network (RNN) architecture used to store and filter information better than standard RNNs.

For English language generation, we selected character-based recurrent neural network model. We use a corpus created from collection of articles from wikipedia, more discuss in dataset section. In section 2 we will discuss dataset and preprocessing. In Section 3, we will talk about structure of the model and its layers. Section 4 will tell us about model training and text generation and at last we will talk about evaluation metrics and conclusion.

Related Work

Text generation task is performed through different approaches like logical approach, statistical process, and neural networks. Leblet et al. (2003) [14] use logical approach in which ROBDD Boolean function and SVM as text classifier is used to generate queries. Francois et al. (2010) [13] used statistical process for language generation. Dynamic Bayesian network BAGEL (Bayesian networks for generation using active learning) is used for statistical language generator. Yang, et al. (2020) [15] proposed a generative adversarial network (GAN) based model for text generation. Hochreiter et al. (1997) [6] introduce the Long Short Term Memory (LSTM) cell in recurrent neural network for the first time and now it is widely used for text generation. Dialogue generation system is build using LSTM-RNN by Wen et al (2015) [7]. Lifeng et al.(2015) [12] proposed RNN based model in encoder-decoder framework to generate response for conversation system.

II. Dataset and preprocessing

To generate wikipedia like text we need to train our model on a dataset of wikipedia articles. We have used a dataset of wikipedia articles developed by non-profit Wikimedia Foundation and it is available on TensorFlow official website. A free software MediaWiki is used to collect the articles from the wikipedia website. The dataset contains 6,033,151 cleaned articles of all languages with total size of 33.77 GiB. In the cleaning process the unnecessary parts of the articles like tables, references, pictures, expressions, and graphs are filtered out. To work quickly and efficiently dataset is converted into TensorFlow dataset format known as tfds. TensorFlow Dataset tfds is a convenient way to download and load large size datasets into machine learning frameworks.

A. Flatten the dataset and generate vocabulary

We need only the text so heading, and titles are filtered out of each articles and convert the string of text into sequence of characters as we have to use a character-based Recurrent

Neural Network (RNN) to generate the text. After converting the text into more granular characters dataset, each single dataset item represents a single character instead of big piece of text. In the next step, we have generated the vocabulary which contains each unique characters of the dataset.

B. *Vectorize the characters*

Next we use vocabulary to vectorize the characters. By using the generated vocabulary we convert the sequence of characters into sequence of vectors, so we can feed the sequence of the character as input to the recurrent neural network. Here we also define the maximum length of sequence of characters to input the model. Following parameter shown in table 1 are used to shuffle and split the character sequences into manageable input sequences.

TABLE I.

Parameter	Value
Input_Sequence_Length	200
Target-Sequence_Length	200
Buffer_Size	100
Batch_Size	64
Prefetch_Size	10

III. Model

We proposed character-based Recurrent Neural Networks (RNN) to generate text. A python deep learning API Keras is used to build the model. Instead of shared layers model we lean towards linear stack of layers. Keras Sequential function is used to build three layers model.

First layer is **Embedding Layer** which is also known as input layer. In this layer lookup table is used to map characters to vectors with embedding dimensions. Embedding layer takes batch of characters indices sequence as an input. Then it encodes each character of the sequence to a vector of selected embedding length.

Second layer is **LSTM Layer** which use a type of Recurrent Neural Network (RNN) famous for memorizing long term dependencies known as Long-Short Term Memory (LSTM). Last layer is **Dense Layer** which is also called output layer

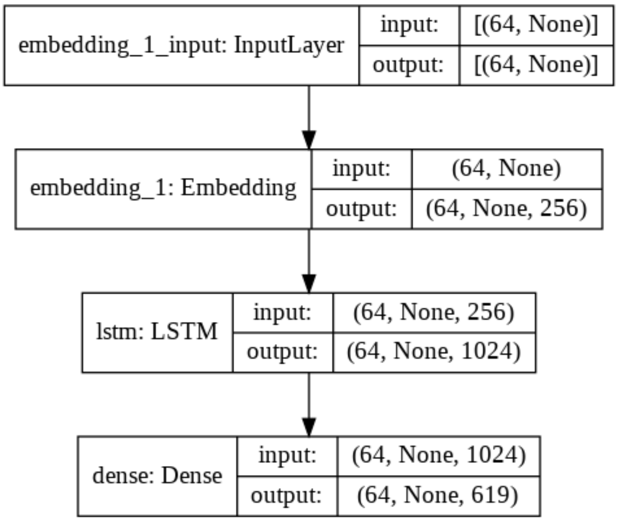

Fig 1 Model Layers

IV. Model Training

A batch of character sequence given as input to the embedding layer. Then the embedding layer lookup the embedding of each characters from the table then embedding layer output the character embeddings which is used as input to the LSTM model. The LSTM model executes one timestep and gives output then dense layer is applied on that output to generates logits. At last logits is used to predict the log-likelihood of the next character. This is treated as a classification problem as predict the class of the next character from given input and previous LSTM internal state.

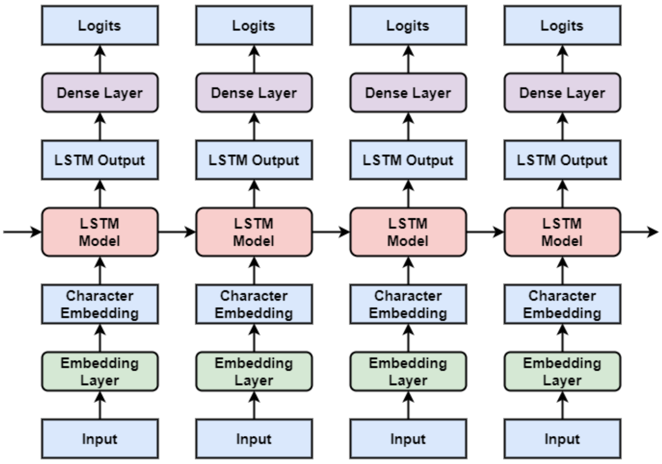

Fig 2 Model Architecture

A. *Loss Fuctions*

We have used Cross-Entropy as loss function as it is a convenient for such model which have timesteps. Basically cross-entropy is used to measure difference between two probabilities distribution for set of timestep events. Cross-Entropy mathematically represented as

$$H(p,q) = -\sum_{\forall x} p(x)\log\left(q(x)\right)$$

where $p(x)$ is actual probability distribution and $q(x)$ is estimated probability distribution. For neural network it can also be represented for single example as

$$L = -y \cdot \log\left(\hat{y}\right)$$

Where the term y is actual vector, $\hat{y}$ is estimated vector and $\cdot$ is inner product

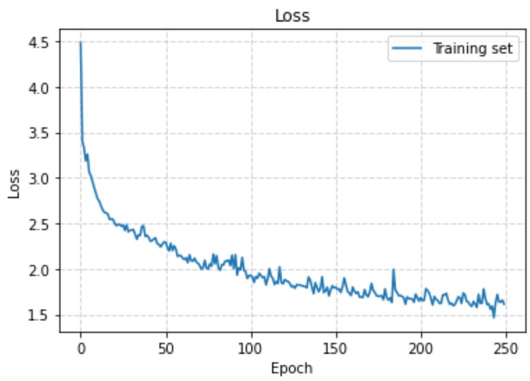

Fig 3. Loss vs Epoch Graph

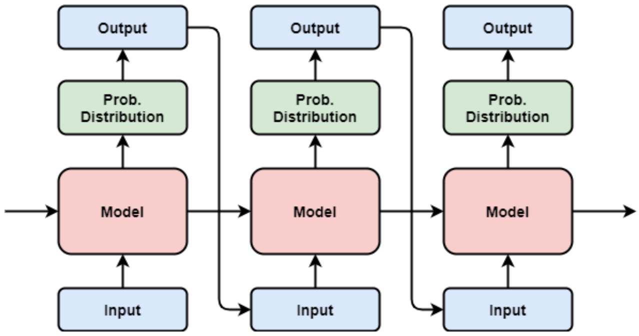

Fig 4 Test Generation Loop

B. Adam Optimizer

We have used a slightly new optimizing algorithm known as Adam[11]. It is very easy to implement and computational efficient. Adam optimizer is commonly used for large datasets with high dimensional parameter spaces[11]. Adam algorithm combines the advantages of previous two popular techniques Aderid and RMSProp. This algorithm updates the weights of the network by following update rule

$$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \frac{\widehat{m}}{\sqrt{\widehat{v_t}} + \epsilon}$$

where $\widehat{m_t}$ is first moment estimate and $\hat{v}_t$ is second raw moment estimate.

C. Text Generation

To generate the text, first of all we need to initialize the model and then select the target sequence length and input sequence of characters. The input character and the RNN state is used to predict the next character. A categorial distribution is used to calculate the index of the predicted character then this character is used as input in to the model in the next timestep. The RNN model fed back the internal state to understand the more context of the character shown in fig 4.

TABLE 2. Text Generation with temperature 0.2

Input	Text Generation
At the beginning	At the beginning of the first season of the population of the state
Science is	Science ist and the constitution of the same controlled in the company of the control of the property of the store
Event	Eventer of the Armenian politician and a construction
Up to date	Up to date and the state of the control of the station of the

TABLE 3. Text Generation with temperature 0.4

Input	Text Generation
At the beginning	At the beginning on became a track and the state in the southern
Science is	Science ist of the state of the first political described in the survival called the murder of the country of the state of the formerly group
Event	Event in 1998 and a main as the support of the 1970s
Up to date	Up to date and were the good at the Parish of the South America

TABLE 4. Text Generation with temperature 0.6

Input	Text Generation
At the beginning	At the beginning in the first parish players by Armenian association
Science is	Science is-motes to addition of the region of Bang Station singles that party of the large-coloration of Archi
Event	Event in the Australian labors and was town coaches modernists to representation in 2011
Up to date	Up to date provides to reaching them the support of the country

TABLE 5. Text Generation with temperature 0.8

Input	Text Generation
At the beginning	At the beginning in June 1972, June's national, to early currently
Science is	Science is of its layers in 2015. The consess in November 1947 by sport match was a women's Business alreased
Event	Event by, a partially and thing. In 1978, it was a song may a bass one that they very of the punnake show in 2005.
Up to date	Up to date, as a former architecture encourages) to the median advanches of the comply of American loss a state

TABLE 6. Text Generation with temperature 1.0

Input	Text Generation
At the beginning	At the beginning [Robberal Soccer Healton. (2013) "(John 1846) was
Science is	Science is—on Get in 710-Thanaki in 135 August 24, Ind, Minerington jained Team
Event	Eventressen, in 0 four Bymanian players (in the concelling and would bodford pays suggesion) in assescility simmetition by the hanged above 5.1.
Up to date	Up to dates Kernadk in the tickers arts a carpears, allower

D. Evalaution

The model is evaluated on two metrics BLEU[10] and BERTScore[8]. The Bilingual Evaluation Understudy (BLEU) is the first metrics which is used to measure similarity between candidate and reference sentences. BERTScore is the other evaluation metric to evaluate the model. Originally the BERTScore metric is used for text generation[8] and BLEU metric is used for machine translation[10] but also commonly used for text generation[9]. BERTScore is used to measure the similarity between the candidate and reference sentence with respect to semantics. Results in table 7 shows that model outperforms many models includes statistical and logical models.

TABLE 7 Evaluation

Temperature	**BERTScore**			**BLEU**
	P	**R**	**F**	
0.2	0.8609	0.8629	0.8619	0.74
0.4	0.6862	0.8652	0.8617	0.66
0.6	0.8438	0.8600	0.8518	0.66
0.8	0.8193	0.8443	0.8316	0.56
1.0	0.8010	0.8102	0.8204	0.54

V. Conclusion

We proposed a LSTM character-based language model for text generation. We used a dataset which is a collection of English articles from wikipedia. Model is implemented through tensorflow framework. Cross Entropy is used as loss function and Adam Optimizer algorithm is used to optimize the model. The text generation is then evaluated by using BERTScore and BLEU metrics. The model gives promising result as comparing to other models including logical models and statistical models. In the future we would like to fine tune the parameters to get more precise results.

References

[1] Guthaus, M. R., Ringenberg, J. S., Ernst, D., Austin, T. M., Mudge, T., & Brown, R. B. (2001, December). MiBench: A free, commercially representative embedded benchmark suite. In Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on (pp. 3-14). IEEE.

[2] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112).

[3] Nallapati, R., Zhou, B., Gulcehre, C., & Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. arXiv preprint arXiv:1602.06023.

[4] Sutskever, I., Martens, J., & Hinton, G. E. (2011). Generating text with recurrent neural networks. In Proceedings of the 28th International Conference on Machine Learning (ICML-11) (pp. 1017-1024).

[5] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. Neural Computation, 9(8):1735–1780, 1997.

[6] S. Hochreiter and J. Schmidhuber, ''Long short-term memory,'' Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[7] T. H. Wen, M. Gasic, and N. Mrksic, ''Semantically conditioned LSTM-based natural language generation for spoken dialogue systems,'' Comput. Sci., vol. 3, no. 17, pp. 144–152, 2015.

[8] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, Yoav Artzi (2020) BERTScore: Evaluating Text Generation with BERT. arXiv:1904.09675v3

[9] Asli, C, Elizabeth, C, Jianfeng, Gao. (2020). Evaluation of Text Generation: A Survey. arXiv:2006.14799

[10] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. Computational Linguistics. Association for Computational Linguistics, 2002.

[11] Diederik, P. Jimmy, L. (2017) ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. arXiv:1412.6980v9

[12] Lifeng Shang, Zhengdong Lu, Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. arXiv:1503.02364v2

[13] Francois Mairesse, Milica Gasic, Filip Jurcicek, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. Phrase-based Statistical Language Generation using Graphical Models and Active Learning Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 1552–1561, 2010.

[14] Leblet Jimmy and Quafafou Mohamed. 2003 A new method for query generation applied to learning text classifiers. Proceedings of the IEEE/WIC International Conference on Web Intelligence (WI'03) 0-7695-1932-6/03

[15] Y. Yang, X. Dan, X. Qiu and Z. Gao, "FGGAN: Feature-Guiding Generative Adversarial Networks for Text Generation," in *IEEE Access*, vol. 8, pp. 105217-105225, 2020.