ISSN No:-2456-2165

Bidirectional LSTM Networks for Poetry Generation in Hindi

Ankit Kumar
University of Delhi
New Delhi, India

Abstract:- This paper proposes a self-attention enhanced Recurrent Neural Networks for the task of poetry generation in Hindi language. The proposed framework uses Long Short-Term Memory with multi-head self-attention mechanism. We have utilized the multi-head self-attention component to further develop the element determination and hence protect reliance over longer lengths in the recurrent neural network architectures. The paper uses a Hindi poetry dataset to train the network to generate poetries given a set of words as input. The two LSTM models proposed in the paper are able to generate poetries with significant meaning.

Keywords:- *Hindi Poetry Generation, Text Generation, Poetry Generation, Long Short-Term Memory.*

I. INTRODUCTION

From short stories to composing a great many expressions of books, machines are producing words more than ever. There are huge loads of models accessible on the web where designers have utilized AI to compose bits of text, and the outcomes range from the silly to magnificently interesting. On account of significant progressions in the field of Natural Language Processing (NLP), machines can comprehend the unique situation and twist up stories without help from anyone else. Instances of text generation incorporate machines composing whole parts of famous books like *Game of Thrones* and *Harry Potter*, with varying levels of accomplishment. For instance, Daza et al., [1] used texts from Homer's *Odyssey* to George R. R. Martin's *Game of Thrones* to generate stories.

An enormous piece of information that we experience is text based. Text information requires considering both the semantic and syntactic importance of words. With the methodology of deep learning, Natural Language Processing (NLP) has achieved new heights. It enables our machines to examine, grasp and decide significance out of writings. Recurrent Neural Network (RNN) has appeared as an encouraging option to withstand the test of time on various text-based tasks.

Recurrent Neural Networks have been used for various applications like text classification [2], language translation [3], image captioning [4], speech recognition [5], and numerous others. Hypothetically, vanilla Recurrent Neural Networks exhibit dynamic worldly conduct for a time series task. Be that as it may, as clarified by Hochreiter [6] and Bengio et al., [7] vanilla Recurrent Neural Networks are powerless to dispersing or exploding slopes. To beat this restriction, Hoschreiter, in 1997 presented Long Short-Term Memory (LSTM) [8]. LSTM utilizes three entryway system namely- forget, input and output gates to tackle the slope issue.

The rest of this paper is structured as follows: Section 2 highlights some of the work being carried out in the field of text generation using RNN and the calculation of self-attention mechanism. Section 3 details our model, while section 4 incorporates the details of our implementations, datasets, results and the observations made based on the basis of the outcomes of our experiments. We conclude this paper with our final remarks in section 5. The remainder of this paper is organized as follows: Section 2 features some of the work being completed in the field of text generation utilizing RNN and the estimation of self-attention mechanism. Section 3 subtleties our model, while section 4 consolidates the subtleties of our executions, datasets, results and the observations made based on the results of our analyses. We conclude this paper with our final remarks in section 5.

II. BACKGROUND

A. Recurrent Neural Networks for Text Generation

Recurrent neural network is a sequential network wherein yield at each progression is the capacity of its present information and the yields of the past inputs. With the new advancement in the field of text generation utilizing RNN, recurrent networks are presently being utilized for an assortment of undertakings in text generation. Fedus et al., [9] used RNNs in GAN network for text generation by filling in the slot. Pawade et al., [10] used RNN for training a story scrambler and Abujar et al., [11] used RNNs for training a model to generate text in Bengali language. RNNs are being utilized for different undertakings in text generation. It appears to be simply rational to investigate more with RNN for the task of text generation.

B. Long Short-Term Memory

Long Short-Term Memory (LSTM) networks have proved over the years that they can resolve the issue of vanishing gradients in Recurrent Neural Networks. The LSTM design comprises of memory blocks which are basically a bunch of recurrently connected subnets. Each block contains at least one self-associated memory cell and three multiplicative units - the input gate, the output gate and the forget gate. A LSTM network is framed precisely like a basic

RNN, then again, actually the nonlinear units in the hidden layers are supplanted by memory obstructs.

Given an input sequence x = (x1, ..., xn), a vanilla recurrent neural network (RNN) computes the hidden vector sequence h = (h1, ..., hn) and output vector sequence y = (y1, ..., yn) by repeating the following equations from t = 1 to n:

$$h_n = \mathrm{F}(W_{xh}x_n + W_{hh}h_{n-1} + b_h) \qquad (1)$$

$$y_n = W_{hy}h_n + b_0 \qquad (2)$$

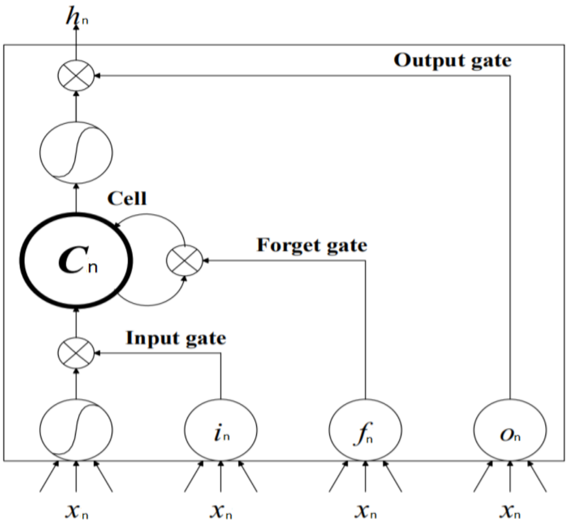

Fig. 1. An LSTM cell with input (i_n), output (o_n) and forget (f_n) gates, and h_n the LSTM cell output.

where the W terms denote the weight matrices, the b terms denote the bias vectors and F denotes the hidden layer element-wise application of a sigmoid function. However, we have observed that LSTM solves the vanishing gradient problem. Therefore, instead of calculating F based on vanilla RNN network, we are going to calculate using LSTM equations as following.

$$i_n = \sigma(W_{xi}x_n + W_{hi}h_{n-1} + W_{ci}c_{n-1} + b_i)$$
(3)

$$i_n = \sigma(W_{xf}x_n + W_{hf}h_{n-1} + W_{cf}c_{n-1} + b_f)$$
(4)

$$c_t = f_n c_{n-1} + i_n \, tanh(W_{xc}x_n + W_{hc}h_{n-1} + b_c)$$
(5)

$$o_n = \sigma(W_{xo}x_n + W_{ho}h_{n-1} + W_{co}c_n + b_o)$$
(6)

$$h_n = o_n \, tanh(c_n)$$
(7)

where σ is the logistic sigmoid function, i is the input gate, f is the forget gate, o is the output gate. i, f, o and c together form the cell activation vectors. The size of these vectors is same as the hidden vector h. W_{hi} is the hidden-input gate matrix, W_{xo} is the input-output gate matrix. The weight matrix from the cell to gate vectors are diagonal, so element t in each gate vector only receives input from element t of the cell vector. The bias terms have been overlooked for clarity.

C. *Bidirectional RNNs*

One shortcoming of vanilla RNN is that it uses previous contexts only. Bidirectional RNNs (BRNNs) overcome this by processing the data in both the forward and the backward direction before being fed to output layer. Bidirectional RNNs calculate the forward hidden sequence, the backward hidden sequence and the output sequence y by iterating the backward layer from n = T to 1, the forward layer from n =1 to T and then updating the output layer y_t:

$$\overrightarrow{h_n} = \mathrm{F}(W_{x\vec{\mathrm{h}}}x_n + W_{\vec{\mathrm{h}}\vec{\mathrm{h}}}\overrightarrow{h_{n+1}} + b_{\vec{\mathrm{h}}}) \qquad (8)$$

$$\overleftarrow{h_n} = \mathrm{F}(W_{x\overleftarrow{h}}x_n + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h_{n+1}} + b_{\overleftarrow{h}}) \qquad (9)$$

$$y_n = W_{\vec{\mathrm{h}}y}\overrightarrow{h_n} + W_{\overleftarrow{h}y}\overleftarrow{h_n} + b_y \qquad (10)$$

Combining Bidirectional RNNs with LSTM allows the network to access long-range context in both the directions.

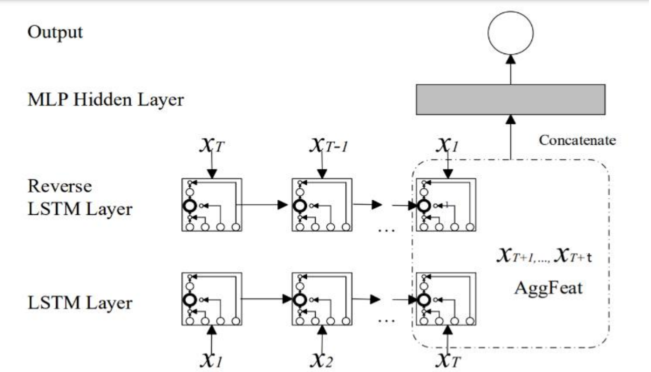

Fig. 2. A Bidirectional RNN with forward and backward layers.

D. *Self-Attention*

There are several attention models proposed starting from Bahdanau's attention model [12] to Transformer [13]. The attention mechanism permits output to pay attention on input while delivering output while the self-attention model permits interactions on inputs with one another i.e. compute attention of all other inputs with respect to one input. Poetry involves paying attention to all the words. Therefore, in this work we proceed with self-attention mechanism proposed by Lin et al., [14] in 2017.

Self-attention sublayers utilizes h attention heads. To frame the sublayer yield, parameterized linear transformation is applied to the concatenation of results from each head. Each attention head computes a new sequence z = (z_1, z_2, ..., z_n) by operating on the input sequence x = (x_1, x_2, ..., x_n).

$$z_i = \sum_{k=1}^{n} \alpha_{ik}(x_k W^V) \qquad (11)$$

Each weight coefficient, α_{ik}, is computed using a softmax function:

$$\alpha_{ik} = \frac{exp\ e_{ik}}{\sum_{j=1}^{n} exp\ e_{ik}} \qquad (12)$$

Further, e_{ik} is calculated using a function that compares two inputs:

$$e_{ik} = \frac{(x_i W^Q)\ (x_k W^j)^T}{\sqrt{d_z}} \tag{13}$$

W^V , W^Q, W^J ε $R^{d_x \times d_z}$ are matrix parameters. These matrices are unique for each attention head and layer.

III. MODEL DETAILS

In this work, we have experimented with two models: a Bidirectional LSTM Generator with Self-Attention (BLG-SA) and without Self-Attention (BLG), and a Bidirectional LSTM-Conv Generator with Self-Attention (BLCG-SA) and without Self-Attention (BLCG).

A. Bidirectional LSTM Generator

BLG use the Bidirectional Recurrent Neural Network with LSTM cells for generating the poetry in Hindi.

B. Bidirectional LSTM Generator with Self-Attention

BLG-SA use the Bidirectional Recurrent Neural Network with LSTM cells. At the top of this architecture the self-attention mechanism is applied to ensure better focus on each word with respect to every other word in the input.

C. Bidirectional LSTM-Conv Generator

BLCG use the combination of Convolution Neural Network in combination with a Bidirectional Recurrent Neural Network with LSTM cells for generating the poetry in Hindi.

D. Bidirectional LSTM Conv Generator with Self-Attention

BLCG use the combination of Convolution Neural Network in combination with a Bidirectional Recurrent Neural Network with LSTM cells. At the top of this architecture the self-attention mechanism is applied to ensure better focus on each word with respect to every other word in the input.

IV. EXPERIMENTS

A. Dataset

We have trained and tested our model using the poetries in Hindi scraped from various websites. The dataset consists of poetries of Gulzar and Rahat Indori. There are 5409 lines of poetries used for training the model and 2318 lines of poetries for validating the model.

B. Implementation

The training data is firstly preprocessed before it is used for training the models. Firstly, we have split the data at the occurrence of next line so that each line of data could be used as a sequence of training data. After splitting the rows, we tokenize our data and prepare the input sequence using the list of tokens. Once the input sequence is prepared, we pad all the sequence to be of equal length with their length being equal to the length of maximum length sequence (11). Also, we modify our labels using one-hot-encoder.

Bidirectional LSTM Generator is the first model that we implement in this work. The model has 100 LSTM cell based Bidirectional Recurrent Neural Network. Before we feed the data to BLSTM, the data is sent to the embedding layer which outputs a (None, 10, 100) dimension sequence which is fed to the BLSTM. The output of BLSTM is trained through a dense layers with 2222 cells before being fed to softmax layer with 1111 cells. Similarly we implement the BLG-SA model also where we add the self-attention layer after LSTM to ensure better interaction of each cell with all other cells.

The third model we implement is the BLCG model which has the embedding layer which outputs (None, 10, 64) dimension output. This sequence is then fed to the Bidirectional LSTM with 64 LSTM cells in each direction. We also use 20% dropout here to prevent overfitting in the network. The output from LSTM is fed to a Convolution Neural Network with 128 one-dimensional convolutions with relu activation. The output of this layer of CNN is fed to the pooling layer where we use max pool with pool size of 4. The output of CNN is again fed to a 128 cell LSTM layer which in feeds its output to a dense layer with l2 regularization which in turn feeds its output to the softmax layer. The BLCG-SA model adds the self-attention layer after the 128 cell LSTM layer and the BRNN layer. Self-attention layer uses sigmoid activation.

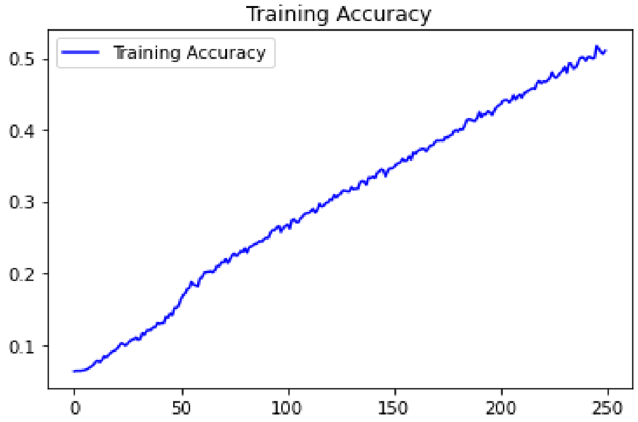

Fig. 3.Training accuracy for Bidirectional LSTM Self-Attention model.

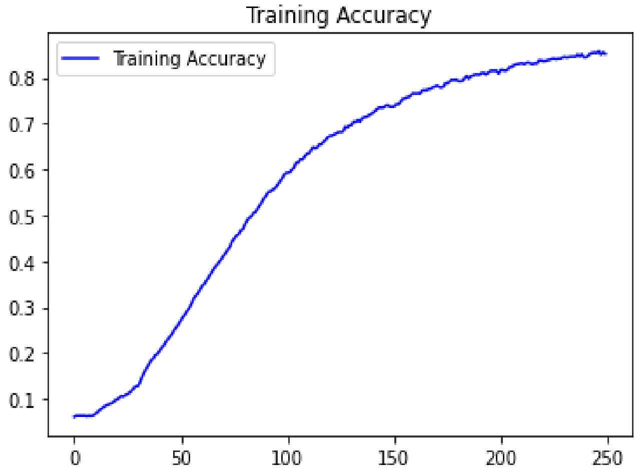

Fig. 4. Training accuracy for Bidirectional LSTM Convolution Generator with Self-Attention model (BLCG-SA).

All the models use categorical cross entropy as the loss function, Adam optimizer and a learning rate of 0.001. While experimenting, we realized that since it is the task of text generation, therefore we probably don't need testing because no matter how much we try the output will always be something new and hence untestable.

C. Results

The results of all the models emphasize that the models are able to learn the pattern and are able to generate poetries that have some meanings. Some of the poetries generated by the model were surprisingly too good from what we expected. The Bidirectional LSTM with Convolution Layer and Self Attention layers is able to generate the most better poetries while the poetries generated by the basic Bidirectional Recurrent Neural Network with LSTM cells were lesser meaningful almost all the times as compared to other models. The BLCG model was able to output better poetries then the self-attention based BRNN model aka BLG-CA model.

D. Observations

We make following observations from the set of experiments conducted in this work:

- Self-attentional models perform better than their non-self-attention based parent models.
- The LSTM-CNN based models are able to learn the sequences better than LSTM alone.
- The LSTM-CNN based models have lesser number of parameters hence the training time is lesser than for comparable LSTM network.
- Self-attention when added to LSTM networks, increases the quality of learning for the network.

V. CONCLUSION

In this paper, we have done a series of experimentation with Bidirectional RNN architecture using LSTM cells with and without self-attention and also in combination with Convolution Neural Network. Our results show that Bidirectional RNN with LSTM cells in combination with CNN are able to learn the sequences better and with lesser training time also. At the same time, the basic bidirectional LSTM models is also able to learn qualitative enough sequences and is able to give outputs which are meaningful.

REFERENCES

[1]. Daza, Angel, Hiram Calvo, and Jesús Figueroa-Nazuno. "Automatic text generation by learning from literary structures." In Proceedings of the Fifth Workshop on Computational Linguistics for Literature, pp. 9-19. 2016.

[2]. D. Tang, B. Qin, and T. Liu, ”Document modeling with gated recurrent neural network for sentiment classification.” In Proceedings of the 2015 conference on empirical methods in natural language processing, pp. 1422-1432. 2015.

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

[4]. O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, ”Show and tell: A neural image caption generator.” In Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on, pp. 3156-3164. IEEE, 2015.

[5]. A. Graves, A. R. Mohamed, and G. Hinton, ”Speech recognition with deep recurrent neural networks.” In Acoustics, speech and signal processing (icassp), 2013 ieee international conference on, pp. 6645-6649. IEEE, 2013.

[6]. S. Hochreiter, ”The vanishing gradient problem during learning recurrent neural nets and problem solutions.” International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6, no. 02 (1998): 107-116.

[7]. R. Pascanu, T. Mikolov, and Y. Bengio, ”On the difficulty of training recurrent neural networks.” In International Conference on Machine Learning, pp. 1310-1318. 2013.

[8]. S. Hochreiter, and J. Schmidhuber, ”Long short-term memory.” Neural computation 9, no. 8 (1997): 1735-1780.

[9]. Fedus, William, Ian Goodfellow, and Andrew M. Dai. "Maskgan: better text generation via filling in the_." arXiv preprint arXiv:1801.07736 (2018).

[10]. Pawade, D., A. Sakhapara, M. Jain, N. Jain, and K. Gada. "Story scrambler-automatic text generation using word level RNN-LSTM." International Journal of Information Technology and Computer Science (IJITCS) 10, no. 6 (2018): 44-53.

[11]. Abujar, Sheikh, Abu Kaisar Mohammad Masum, SM Mazharul Hoque Chowdhury, Mahmudul Hasan, and Syed Akhter Hossain. "Bengali text generation using bi-directional RNN." In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1-5. IEEE, 2019.

[12]. D. Bahdanau, K. Cho, and Y. Bengio, ”Neural machine translation by jointly learning to align and translate.” arXiv preprint arXiv:1409.0473 (2014).

[13]. Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In Advances in neural information processing systems, pp. 5998-6008. 2017.

[14]. Z. Lin, M. Feng, C. N. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, ”A structured self-attentive sentence embedding.” arXiv preprint arXiv:1703.03130 (2017.