RESEARCH ARTICLE OPEN ACCESS

Evaluation of Gated Recurrent Neural Networks on Deep Sentence Classification

Amit Adate [1], ShubhamPathak [2]
School of Computer Science and Engineering
Vit University,Vellore
Tamil Nadu – India

ABSTRACT
Most popular application of Recurrent neural net-works is currently in the field of natural language processing, where sequential data is involved. In this paper we have developed a model that addresses sentence classification, a trending topic in the current natural language processing research, using recurrent neural networks.We use the variants of Recurrent Neural Net-works called as Long Short Term Memory Networks and a quite recently introduced variant Gated Recurrent Units.We have build a model for comparing the both of them. One of major drawbacks with traditional Recurrent networks was that information could not be stored for larger duration of time and to store data for longer time but was not possible in practicality. We also present the graphical analysis of the features learned by our model

Keywords :—Long Short Term Memory, Gated Recurrent Units, Sentence Classification

I. INTRODUCTION

Recurrent neural networks(RNN) are quite useful when we are dealing with sequential data or when we have data of varying length.More recently systems were developed using RNN architecture to develop more complex systems dealing with machine translation.

However not entire progress was made using traditional vanilla RNNs.They were indeed evolutionary but significant advancements in this field was made using more advanced model such as LSTMs where we introduce some advanced recurrent hidden units also called as gates and they were successful.

In this paper we will be seeing the working of two closely related variants of recurrent neural units which are LSTM and its variant called GRU.LSTM and GRU are mostly same however their architecture is different.In GRU we have a slightly reduced architecture compared to LSTM.

II. BACKGROUND:LSTM

LSTM version which is more commonly used in literature was described by Graves and Schmidhuber(2005).The fun-damental idea behind LSTM was use of memory cells,for retaining data for longer time and overcoming the limitations of Recurrent neural networks.RNNs have problem with rec-ognizing long term dependencies like recognizing sequences of words which are quite apart from each other,this prob-lem is also referred to as vanishing gradient problem.More technically speaking the values in the matrix and multiple matrix multiplication are diminishing or becoming closer t

zero and after few time steps they vanish completely[2].At far away time steps gradient is zero and these gradients are not contributing to learning.In fact vanishing gradient problem is not only limited to RNN.They are also observed in case of deep feed forward neural networks.They are very common in RNNS.But fortunately to tackle with these problems LSTMs and GRU architectures were developed.Later on we will be seeing in results how LSTM and GRU are able to deal with vanishing gradient problem and help in learning the long term dependencies.Lets describe the LSTM architecture in following diagram.

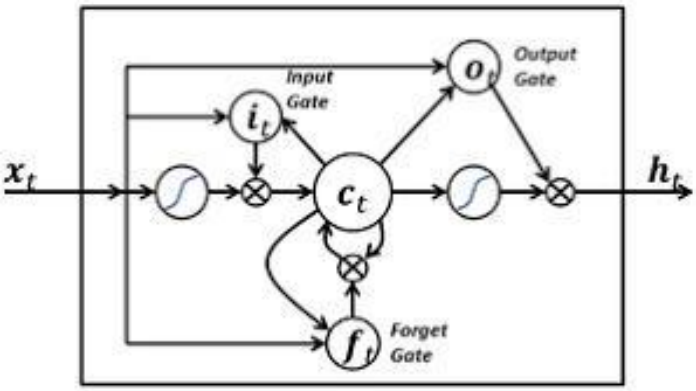

Figure 1. Long Term Short Term Memory

The equations below describe how layer of memory is updated is updated at every time step t.In these equa-tions: x_t is the input to the memory cell at time t. $W_i,W_f,W_c,W_o,U_i,U_f,U_c,U_o$ and V_o are weight matrices. b_i,b_f,b_c,b_o are the bias vectors.
First we describe the value of input gate i_t,C_t candidate values for state of memory cells at time t.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$
$$\widetilde{C_t} = tanh(W_c x_t + U_c h_{t-1} + b_c)$$

Second we compute the activation for forget gate f_t at time t:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

Now we can compute candidate cells new value C_t at time t:

$$C_t = i_t * \widetilde{C}_t + f_t * C_{t-1}$$

With the new state of the memory cells, we can compute the value of their output gates and, subsequently, their outputs:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_o C_t + b_o)$$
$$h_t = o_t * tanh(C_t)$$

An LSTM consists of three gates,first there is a sigmoid forget gate which outputs the value between 0 and 1.A state of zero represents completely get rid of this while 1 represents completely keep this so the basic function of this forget gate layer is to decide which information to discard.Next we have to decide which information we will be storing in our cell.For this we have two parts,first a sigmoid input layer which decides which are the values that have to be updated,and second is tanh layer which is for creating a new candidate vector that can be added to the state.Finally we need to calculate what we have to output, for this we run a sigmoid layer which decides which things are needed to be output and then we put the cell sate through a tanh function so that the values which we decided in the previous step are output.There are several popular variants such as the one proposed by Gers Schmidhuber (2000) which proposes peephole optimization another interesting theory.

III. BACKGROUND:GRU

GRUs are another modified variant of RNNs precisely they are the variants of LSTM itself with slightly reduced architecture and they were more recently proposed by Cho et al. [2014].Like LSTMs GRUs also use gated mechanism to deal with problem of vanishing and diminishing gradient however unlike LSTMs they don't have separate memory cells .Again we have an activation function and a update gate to calculate the percentage of this activation.In most of the cases the results of LSTMs and GRUs are comparable but in some particular cases we observe that results of GRU are slightly better than LSTM[2] as we will be seeing these observations later on in our paper.Now lets describe the architecture and some equations related to the working of GRU.

$$z = \sigma(W_z h_{t-1} + U_z x_t)$$
$$r = \sigma(W_r h_{t-1} + U_r x_t)$$
$$c = tanh(W_c(h_{t-1} \otimes r) + U_c x_t)$$
$$h_t = (z \otimes c) \oplus ((1 - z) \otimes h_{t-1})$$

IV. DISCUSSION ABOUT THE TWO MODELS(LSTM VS GRU

A common observation is that LSTM and GRUs give comparable results as it is evident from the experimental section below,and for some cases GRU gives slightly better results than LSTM.Basically both these models implement a different gating mechanism to prevent the loss of information by vanishing gradient.In case of GRU there no memory

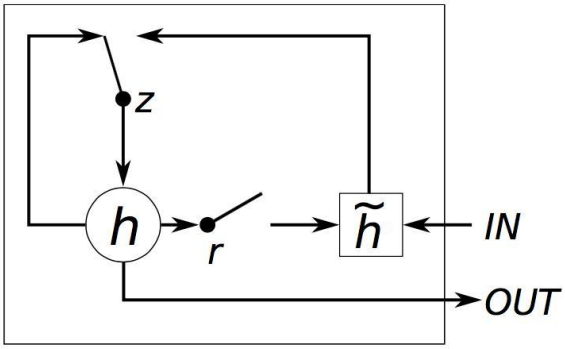

Figure 2. Gated Recurrent Unit Diagram

cells,it simply exposes the full hidden content without any restriction.To precisely say of above two models which model is better we have to keep data sets in mind too as for some specific data sets results observed through GRU is better than LSTM[3].So based on the dataset we simply train both the models and then say about their efficiency or which model is better.Coding wise its simpler to use GRU than LSTM as their architecture is simpler[4],thus easier to modify,computations are also less in GRU as we have only two gates.Also in cases of language modelling where where we have less training data GRUs perform better than LSTMs.Theoretically LSTMs would perform better in case longer-distance modelling sequences.

Although GRUS seem to be quite promising their trade offs are not known as they are quite recent introduced in 2014 and still lots of researches are going on[5].According to empirical evaluations in Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling and An Empirical Exploration of Recurrent Network Architectures, there isn't a clear winner

between the two.[6]

V. EXPERIMENTS

In this section we introduce the model we have implemented for LSTM cells and its variant GRU cells. The primary dataset we experimented upon is Kaggle's IMDB 5000 dataset[7].

We compare the LSTM unit and the GRU unit in the task of sentence classification. We begin by building an LSTM encoder, we intent to encode all data, text format in the last output of RNN before we applying a feed forward network for classification. Our work is philosophically similar to neural transllation machine and sequence to sequence learning ref*

We use an LSTM layer in keras to implement the seq-to-seq model, other than that we also use the GRU cell to build up a similar layer. We use a bi-directional LSTM layer and concatenate the last output of both the layers. Keras provides a nice wrapper called bidirectional, which supported the buildup

of the model. Also, we have implemented an attention layer which is a similar implementation from these papers [8][10].

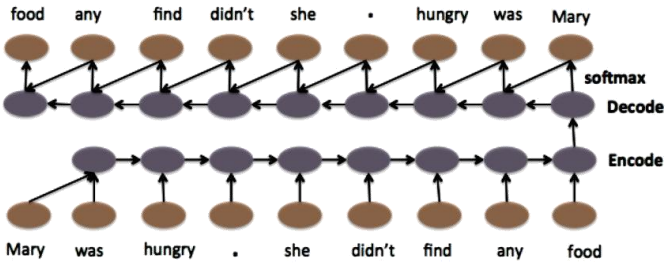

Figure 3. Representation of the model implemented

The vector in the hidden sequence are fed learnable functions to produce an additional vector, probability vector . The output vector is computed as a weighted average of all the hidden state sequences, with the weighting given by .

In our implementations we also added bias unitsb,c. For that we needed to change the nationalizations of our input parameters because they now have different sizes. We have not shown the implementation here, it is available on this repository[9].

VI. RESULTS

We trained a small model very similar to the one here [8]. Our vocabulary size of 8000, mapped into 48-dimensional vec-tors, and used two 128-dimensional GRU layers. The training occured in bathes of 2000 samples, with a validation set of 5000 samples. To achieve best performances, we performed fine tuning of the hyperparameters, tried to improve our inital text processing and also implemented a dropout layer

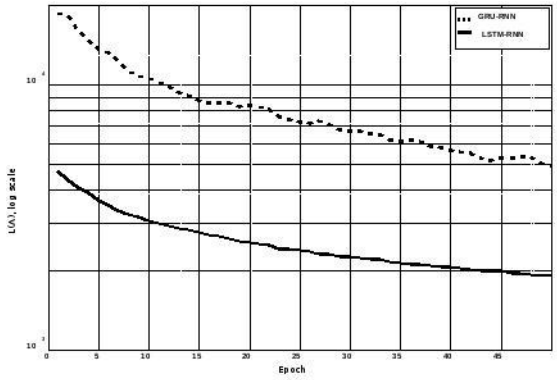

Figure 4. Logarithmic Training Cost vs No of Epochs

Also we briefly experimented on an another dataset of publically available word vectors trained by [1]. And have found that GRU-Cells gave a slight increase in performace mainly due to having fewer hyperparameters and hence train faster and also need less data to generalize compared to the LSTM-Cells but only when the dataset is not large. But with large data, the LSTM's with higher articulateness lead to better results. The best performance we saw was about 90.4%

VII. CONCLUSION

In this paper, we have presented a detailed analysis of two architectures for learning sentence classification. the GRU unit does not have to use a memory unit to control the flow of information like the LSTM unit. It can directly makes use of the all hidden states without any control. GRUs have fewer parameters and thus may train a bit faster or need less data to generalize. But, with large data, the LSTMs with higher expressiveness may lead to better results.

To view our experiments, refer to this repository [9]. Our analysis of the various variants of recurrent neural networks provides significant analysis between the difference in their performance. We conclude that when we have a relatively small dataset, GRU's tend to do perform slightly better and when there's a larger dataset, LSTMs worked better.

REFERENCES

[1] Empirical Evaluation of Gated Recurrent Neural Networks on Sequence modelling by Junyoung Chung, Caglar Gulcehre, KyungHyun- https://www.sharelatex.com/project/598dcc4ffbb49923a45d0777 Cho, Yoshua Bengio (11 dec,2014).

[2] LSTM : A Search Space Odyssey by by Klauss Greff,Rupesh Kumar Srivastava,Jan Koutnik,Bas R.Steunebrink ,Jurgen Schmidhuber

[3] Understanding LSTM's http://colah.github.io/posts/2015-08-Understanding-

LSTMs

[4] LSTM Explanined https://apaszke.github.io/lstm-explained.html

[5] Gated Recurrent Unit (GRU) for Emotion Classification from Noisy Speech by RajibRana, Julien Epps,Raja Jurdak, Xue Li,Roland Goecke, Margot Brereton, Jefferey Soar

[6] Proceedings.mlr.press/v37/jozefowicz15.pdf by R Jozefowicz.

[7] Kaggle Dataset : IMDB 5000 https://www.kaggle.com/deepmatrix/imdb-5000-movie-dataset

[8] Rafael et al :FEED-FORWARD NETWORKS WITH ATTENTION CAN SOLVE SOME LONG-TERM MEMORY PROBLEMS

[9] REPOSITORY FOR OUR CODE : https://github.com/amitadate/gru-sentence-classification

[10] NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE https://arxiv.org/pdf/1409.0473v7.pdf

[11] Analysis of Gated Recurrent Neural Networks on Sequence Modeling https://arxiv.org/pdf/1412.3555v1.pdf