Self-Attention on RNN-based Text Classification

Zeying Tian zeying.tian@mail.mcgill.ca
Dailun Li dailun.li@mail.mcgill.ca
Yining Duan yining.duan@mail.mcgill.ca

Abstract

Nowadays, many researchers claim that the self-attention mechanism generates a better performance regardless of circumstances. Having doubted such opinion, we examine different models with scalar dot-product self-attention mechanism under various experimental settings to develop a comprehensive understanding of such technique. In conclusion, we have verified that the performance of attention mechanism grows as the dataset input increases in size. In addition, we have observed that our attention layer can impact model's performance negatively on small datasets. Moreover, we demonstrate that the attention mechanism is model-dependent: opposite effects may be obtained on the same dataset with different models.

1 Introduction

Text classification is a classical problem. Traditional approaches, including Naive-Bayes and Bag-of-words model, have been used over a long period. However, most of them treat sentences as unordered set of words, and therefore fail to identify syntactic features. The contemporary approach is to use a recurrent neural network (RNN). RNN loops through the sentence, and hidden states can be passed from layers to layers. Though RNN has been successful, it struggles to memory long-range dependencies(Bengio et al., 1994). To tackle this problem, researchers first developed LSTM(Hochreiter and Schmidhuber, 1997), and then GRU(Chung et al., 2014), and subsequently the self-attention mechanism(Vaswani et al., 2017).

In the encoder-decoder architecture, the attention mechanism acts as a middle layer that combines the hidden states and the encoder outputs to extract key information. This mechanism has been shown to achieve profound empirical results in solving various NLP problems, including machine translation, text classification, etc. Additionally, (Vaswani et al., 2017) has shown that the attention mechanism significantly improves the performance on various models. Since the attention mechanism was invented to target long-range dependencies, we hypothesize it to be more effective on larger datasets. Also, we anticipate the self-attention mechanism may behave differently with different RNNs. To test our hypothesis, we conduct a sequence of experiments with LSTM and GRU on various datasets. In this experiment, we target a specific type of self-attention: scaled dot-product self-attention mechanism.

Our result indicates that on corpora where text length is high, the attention mechanism generally has a positive effect. For example, on the YELP dataset, the attention layer increases the models' performance by 0.64%(LSTM) and 0.03%(GRU); on the IMDb dataset, the attention layer improves the validation accuracy by 0.49%(LSTM) and 1.2%(GRU). However, the attention mechanism can be model-sensitive, especially on smaller datasets. On the TREC dataset, for instance, the attention mechanism improves the performance of LSTM by 1% yet decreases that of GRU by 2.2%.

2 Related Works

In this project, we have implemented a type of attention mechanism, namely the **scaled dot-product self-attention mechanism**, which was proposed in the study by(Vaswani et al., 2017). We notice that there already exist multiple related works(Du and Huang, 2018; Zheng and Zheng, 2019; Sun and Lu, 2020; Radford et al., 2018).

In the study by (Du and Huang, 2018), the researchers have tested attention-based RNNs (LSTM and GRU) on Xinhua corpus and Reuters Corpus. In their conclusions, the researchers have discovered that LSTM can be more suitable than

GRU for datasets with longer sentences; this contradicts with our findings as GRU always surpasses LSTM in our experiments. In the study by (Zheng and Zheng, 2019), the researchers have built several attention layers on top of existing RNNs and produced better results on relatively large datasets (Yahoo and Sogou). They state that the attention layer is effective in extracting key words and subsequently boosts overall performance. As we have adopted a different attention mechanism, we have observed different results. Our results show that attention layer may impact the model negatively when the dataset is small. Some researchers observe similar phenomenon(Radford et al., 2018). Finally, there is a study which experiments on other attention mechanisms(Sun and Lu, 2020), which leaves us with possible future investigations.

3 Methods

3.1 Dataset Description

In this project, we conduct experiments on the **TREC**, **SST**, **IMDb**, and the **YelpReviewPolarity** datasets. In these datasets, there are English texts and corresponding labels. Texts are embedded with `torch.nn.Embedding`, where we use pretrained `GloVe.6B.300d` weights across our experiments. The embedding size for each word is 300. Labels are encoded with one-hot encoding. Information of each dataset can be found in Table 1.

Name	Training / Test	Labels	Avg Length
TREC	5500 / 500	6	10.2
SST-2	6920 / 872	2	19.3
SST-5	6920 / 872	5	19.3
YELP	150000 / 38000	2	133
IMDB	25000 / 10000	2	233.8

Table 1: The Dataset Description Table. The **Avg length** stands for the average number of words in the text of each dataset entries. The **Training/Test** specifies the number of training and test entries.

3.2 Classifiers

In this study, our classifier has potentially three components: the encoder, the attention layer, and a linear classifier on top of the attention layer. In scenarios where we do not have the attention layer, we simply have two layers in our classifier: the encoder and the linear classifier.

For encoders, we have chosen two types of RNNs, **LSTM(bidirectional)** and **GRU(bidirectional)**. Throughout our experiments, we configure to use a dropout rate of 0.6, and a hidden dimension of 500. We use `Adam` as the optimizer with default parameters $\alpha = 1e-3, \beta_1 = 0.9, \beta_2 = 0.999$ and adopt cross entropy loss for loss evaluation. The batch size is set to be 32. As to the number of layers, we have tested both 4 layers and 2 layers.

As to the attention layer, we have adopted a **scaled dot-product self-attention mechanism**(Vaswani et al., 2017). Concretely,

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Q, K, V stand for *queries*, *keys*, and *values* respectively, and d_k is a specific keys dimension(Vaswani et al., 2017).

3.3 Experiment Setup

First, we try various hyper-parameters to obtain optimal GRU and LSTM models. Then, we conduct our experiments on different datasets, with and without the attention layer.

4 Results

4.1 Model Fine Tuning

First, we tune the hyperparameters to find the optimal LSTM and GRU models that fit for the datasets.

We examine the performance of 2-layer and 4-layer LSTM model on the TREC dataset. The results are specified in Figure 1.

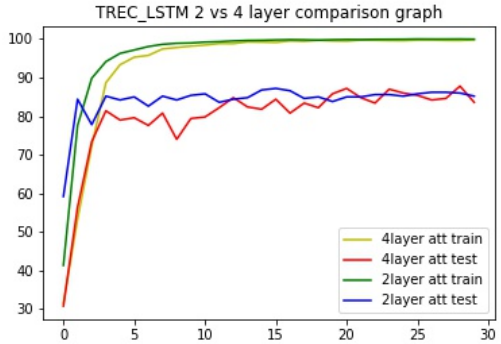

Figure 1: LSTM Performance on TREC with 2 Layers and 4 Layers

Though on the TREC dataset both configurations work similarly, the model with 4 layers fluctuates

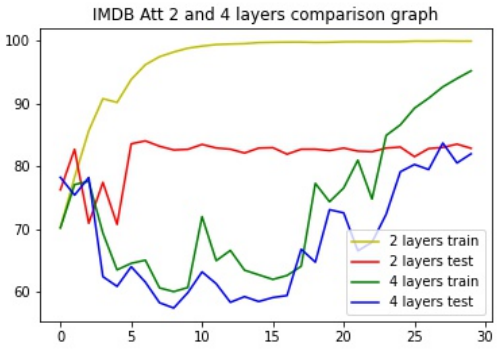

Figure 2: IMDB 2 layer and 4 layer performance comparison

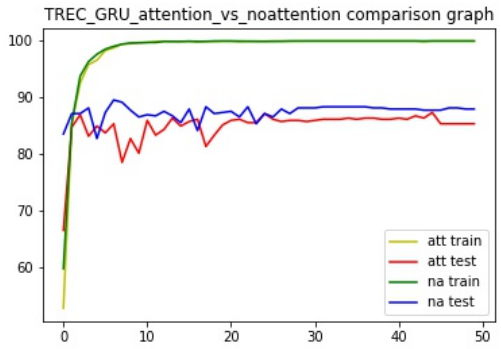

Figure 3: Bidirectional GRU model performance with and without attention

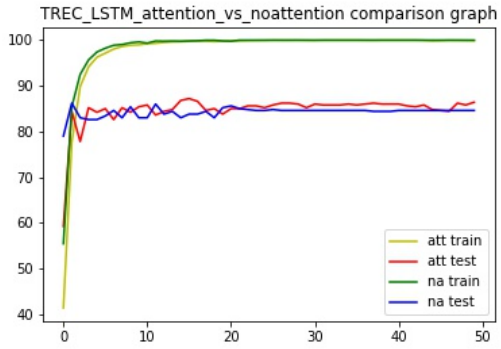

Figure 4: Bidirectional LSTM model performance with and without attention

drastically on the IMDb dataset, as in Figure 2. In order to have consistent results, we decide to use 2 layers for all subsequent investigations.

Moreover, we examine the number of hidden units ranging between 200 and 1000. When the number of hidden units is 200, the model cannot converge and stays around around 55% training accuracy on the IMDb dataset. In contrast, when we have 500 hidden units, the model is able to converge in 30 epochs. Subsequently, we have observed that it takes much longer to train at more hidden units. For example, increasing the number of hidden units from 500 to 1000 on the SST dataset will increase the number of parameters from 11916606 to 37138606, resulting in a roughly 68% decrease in the training speed. Therefore, we choose the number of hidden units to be 500.

4.2 Results Analysis

Our results of LSTM and GRU on all four datasets (with and without the attention layer) are recorded in Table 2. The effects of the attention layer, which are measured by the difference in Top-1 accuracy, are recorded in Table 3.

From Table 2, when implemented without the attention layer, GRU always outperforms LSTM, regardless of datasets. Also, the introduction of the attention layer is not always beneficial and is dependent on the choice of encoders, as shown in Table 3. For instance, when we are using the TREC dataset, the inclusion of attention layer has boosted the performance LSTM by 1.00%, yet has lowered that of GRU by 2.2%. The detailed per-epoch performance on the TREC dataset is depicted in Figure 3 and in Figure 4. Moreover, we have observed a clear trend between the average sentence length of a dataset and the effects brought by attention in the case of GRU. As the average sentence length increases, i.e., going from TREC to IMDb, the improvements brought by attention increase. (-2.2% to 1.2%). However, for LSTM, attention brings moderate improvements across all scenarios. It brings about 1% increase in the accuracy on the TREC dataset, followed by an approximately 0.5% increase on the YELP dataset.

Finally, we can see that attention may be less effective as the number of labels increases, when comparing SST-5 to SST in Table 3.

5 Discussion

First of all, it is clear that attention is useful when the average sentence length is large (such as in the case of IMDb, with average sentence length of 233.8), which confirms our hypothesis. When the average sentence length is large, it is common for RNN to forget long-range dependencies(Bengio et al., 1994). Hence, the introduction of attention mechanism, which is designed to target long-range dependencies, should boost the performance of

Dataset	Model	Attention	Top-1 Val Acc.
TREC	LSTM	Yes	87.2
TREC	LSTM	No	86.2
TREC	GRU	Yes	87.4
TREC	GRU	No	89.6
SST	LSTM	Yes	81.07
SST	LSTM	No	80.6
SST	GRU	Yes	80.8
SST	GRU	No	81.1
SST-5	LSTM	Yes	40
SST-5	LSTM	No	39.78
SST-5	GRU	Yes	39.96
SST-5	GRU	No	41.24
YELP	LSTM	Yes	94.22
YELP	LSTM	No	93.58
YELP	GRU	Yes	94.39
YELP	GRU	No	94.36
IMDb	LSTM	Yes	84.0
IMDb	LSTM	No	83.51
IMDb	GRU	Yes	88.17
IMDb	GRU	No	86.97

Table 2: Top-1 Validation Accuracy LSTM and GRU on Various Datasets

RNNs(Vaswani et al., 2017).

In addition, we have observed that the effects of attention are model-dependent, which also confirms our hypothesis. When tested on SST and TREC, GRU performs worse in the presence of attention, whereas the performance of LSTM is improved noticeably(1.0% on TREC and 0.47% on SST). Another difference lies in the trend between average sentence length and the improvements brought by attention. In the case of GRU, it is clear that attention brings higher improvements as sentence length increases; this relationship does not hold for LSTM, as it receives improvements fairly constantly, regardless of datasets.

One possible explanation of why the attention layer has made GRU perform worse on small datasets is the limitations of our attention mechanism. The dot-product mechanism lacks the ability to adapt its parameter for updating during the training process. As GRU has already shown its robustness on smaller datasets, the introduction of the scaled dot-product attention mechanism may make it more difficult for the model to learn important features. As documented in this study(Sun and Lu, 2020), with parametrized attention mechanism, it is possible to fine tune the attention layer to achieve better results, even on smaller datasets.

Dataset	Model	Improvements by Attention
TREC	LSTM	1.00
TREC	GRU	-2.2
SST	LSTM	0.47
SST	GRU	-0.3
SST-5	LSTM	0.22
SST-5	GRU	-1.28
YELP	LSTM	0.64
YELP	GRU	0.03
IMDb	LSTM	0.49
IMDb	GRU	1.2

Table 3: Improvements on Top-1 Validation Accuracy by Attention

Anther possible explanation of the above phenomena may be the difference between the two RNNs' structures. LSTM does not expose all of its memory units(Hochreiter et al., 2001), whereas GRU does(Chung et al., 2014). For each layer in LSTM, there are two output channels, one for the cell state and one for output. In addition, there are only three gates to modify the cell state, before it is passed on to the next state; this allows LSTM to preserve long-range information, and each layer has limited influence on the cell state. In contrast, in the case of GRU, each layer has only one output channel, which renders its ability to remember long-range dependencies. As such, each layer of the GRU is less influenced by its context but more by its current input. When we add the scaled-product self-attention mechanism, it combines the encoder's outputs and its hidden states to produce the context vector. In the case of GRU, the hidden states, which are more influenced by its current inputs than LSTM, can bring uncertainty if input words varied drastically. Combined with the fact that our attention mechanism is not parameterized, it can be hard for our attention mechanism to remove such local influences, which results in worse performance.

6 Conclusion and Future Investigations

In this study, we have demonstrated the strength and limitations of self-attention mechanism by evaluating the performance under different circumstances. Through the analysis, we discover that in a relatively large dataset, the attention mecha-

nism brings positive improvements. Furthermore, we have verified that our scaled dot-product self-attention mechanism can affect the performance negatively on small datasets. We also empirically compare the GRU with the LSTM models, both with attention mechanism, on a given dataset; the results show that the performance of attention mechanism is model-sensitive.

In the future, we may investigate into parameterized attention mechanisms, such as one adopted in (Sun and Lu, 2020). As such, we can further explore the potentials of attention mechanism. Also, we would like to set up our experiment on larger datasets in terms of both paragraph length and the number of entries.

7 Statement of Contributions

Zeying Tian: Writes and modifies codes to prepare the experiment.
Yining Duan: Designs the experiment and draws graphs for the report.
Dailun Li: Report writing.

8 Appendix

References

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Changshun Du and Lei Huang. 2018. Text classification research with attention-based recurrent neural networks. *International Journal of Computers Communications & Control*, 13(1):50–61.

Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al. 2001. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Xiaobing Sun and Wei Lu. 2020. Understanding attention for text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3418–3428.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Jin Zheng and Limin Zheng. 2019. A hybrid bidirectional recurrent convolutional neural network attention-based model for text classification. *IEEE Access*, 7:106673–106685.

A Appendices

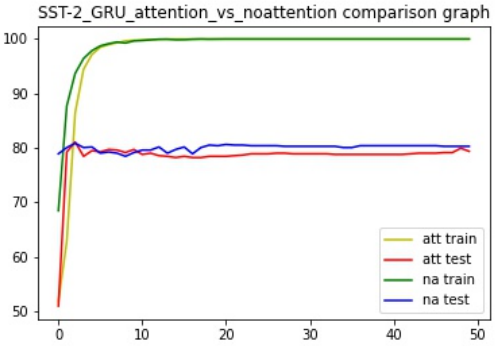

Figure 5

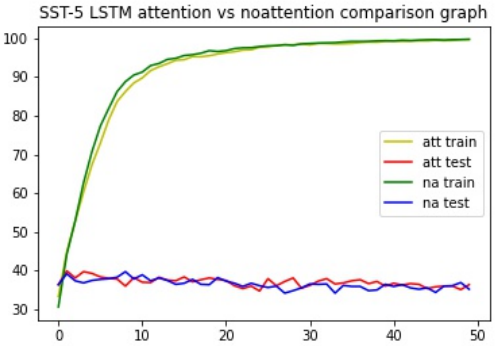

Figure 6

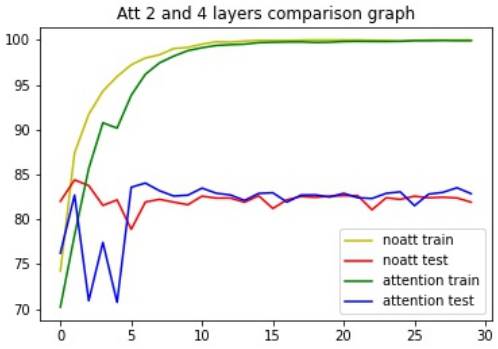
Att 2 and 4 layers comparison graph
100
95
90
85
80
75
70
0
5
10
15
20
25
30
noatt train
noatt test
attention train
attention test

Figure 7