Sentiment Analysis: Transformers and Long Short-Term Memory

Hamin Lee
University of Toronto
haminn.lee@mail.utoronto.ca

Peng-Cheng Chu
University of Toronto
pengcheng.chu@mail.utoronto.ca

Abstract

As most of the general public now express themselves using social media, these platforms have became the key to understanding public sentiments. In this paper, we train different models to learn user sentiment on real world datasets of different sizes. We compare the performance of the typical Recurrent Neural Network model with the ability to learn long-term dependencies (LSTM) and the Transformer model using encoders with bidirectional self-attention mechanism (BERT) on sentiment analysis task for real-world datasets of different sizes.

1 Introduction

Sentiment analysis is the use of Natural Language Processing (NLP) techniques to systematically determine the opinion of a sequential text. It is often used for opinion mining where it determines whether the underlying sentiment of the data is positive or negative [4]. Over the past few years, with abundant data made available through social media, sentiment analysis has been applied extensively to understand the public sentiment.

There have been many breakthroughs in neural network approaches that helped advance the technology for NLP tasks including sentiment analysis. Among all the different NLP techniques, recurrent neural networks have been the main architecture used for sentiment analysis [5].

Recurrent Neural Networks, also known as RNN, is a type of neural network engineered with connections that allow the information of the previous steps to be passed on to later steps. Recently, with more effective models such as Long Short-Term Memory (LSTM) have been used widely in applications involving sequential data.

In this paper, we compare the variations of neural network architectures, namely the Long-Short Term Memory model and the Bidirectional Encoder Representations from Transformers, and their performance on sentiment analysis tasks. We use techniques from publicly available libraries and previous work [7] for pre-processing, word embedding, tokenizing, and training. Furthermore, this paper evaluates the effectiveness of self-attention based Transformer architecture.

2 Related Works

Understanding Recurrent Networks [6]. Recurrent Neural Networks have been used in various applications for sequential data learning tasks. Long Short Term Memory (LSTM) was first introduced by Hochreiter and Schmidhuber [1]. It is a gradient-based approach where the activations of the network are its short-term memory and the weights are its long-term memory [8]. The Gated Recurrent Unit (GRU), introduced by Cho [2], is a variation of LSTM which aims to solve the vanishing gradient problem. In this paper we focus on evaluating the performance of LSTM model on sentiment analysis tasks.

Attention is All You Need [3]. In 2017, Vaswani et al. introduced a self-attention based neural network [3]. The work proposes a new simple network architecture, namely the Transformer. It relies solely on attention-based mechanisms to draw dependencies between inputs and outputs. It allows significantly more parallelization, resulting in a significant increase in training speed. The model extracts features for each word using self-attention, then measures the significance of all the other words in the sentence. In this paper we focus on measuring the performance of Bidirectional Encoder Representations from Transformers (BERT) on sentiment analysis tasks.

3 Methods and Algorithms

3.1 Recurrent Neural Network Models

Long Short-Term Memory (LSTM) Hochreiter and Schmidhuber (1997) introduced LSTM to address the problem of learning long-term dependencies. LSTM mitigates the vanishing gradient problem by encoding the long-term dependencies and relations in a cell state vector. LSTM contains a hidden state h_t^l and a memory vector c_t^l. The form of the update is as follows[6]:

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$
$$c_t^l = f \odot c_{t-1}^l + i \odot g$$
$$h_t^l = o \odot \tanh(c_t^l)$$

Where W^l is a [$4n$ x $2n$] matrix.

3.2 Self-Attention

Transformer Architecture Vaswani (2017) [3] introduced the first fully-attentional Transformer architecture which utilizes the self-attention mechanism [4]. It has shown its effectiveness in language translation tasks on neural machine translation (NMT) [4]. Recently, the application of self-attention networks has expanded to text summarization [10], image classification [12], and sentiment analysis tasks [11].

The two most commonly used attention functions are additive attention and dot-product attention [3]. Among the two, dot-product is much faster and more space-efficient in practice and utilizes the efficiency matrix operation code [3]. "Scaled Dot Product Attention" consists of input queries Q, keys K, values V, and dimension d_k. We compute the matrix of outputs as follows [3]:

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

In this paper, we make use of PyTorch for our Transformer architecture code. The code is based on the paper "Attention is All You Need" [3].

4 Experiments

4.1 Tasks and Datasets

We experiment the performance on sentiment analysis task of the LSTM and Transformer models. The experiments are performed on two datasets of different sizes: the PyTorch internal IMDB dataset which consists of 50,000 movie reviews, each marked as being positive or negative, and the publicly available Twitter dataset on Kaggle with sentimental labels ranging from -1 to 1.

We extracted 5,000 tweets from the Twitter dataset for the purpose of comparing each model's performance on different dataset sizes and processed the data by labelling the negative reviews as 0 and positive reviews as 1.

4.2 Code

The code for the experiement was adapted from `https://github.com/bentrevett/pytorch-sentiment-analysis`. We have forked a publicly available repository and modified it to suit our needs. The code used to produce the results of this paper can be found at `https://github.com/haminthecoder/pytorch-sentiment-analysis` [7].

4.3 Models

In this experiment we will focus on comparing two different models for sentiment analysis. The two models are Long Short-Term Memory[1] and the Transformer[3] model.

For each task, we train two models: a bidirectional LSTM model and a pretrained Bidirectional Encoder Representations from Transformers (BERT) model. In the LSTM model, instead of randomly initializing the word embedding, we use the pretrained GloVe embedding vectors `glove.6B.100d`. For the BERT model, we use the pretrained `bert-base-uncased` tokenizer from the transformer library, which tokenizes the data in a way that is consistent with the pretrained model.

Both models are trained with the Adam optimizer [15]. Adam adapts the learning rate for each parameter, giving parameters that are updated more frequently lower learning rates and parameters updated infrequently higher learning rates [7]. PyTorch is used to construct the optimizer in our model.

5 Result

5.1 Model Comparison

For the sentiment analysis tasks, we split the data for training, validation and testing. Then we train for 8 and 10 epochs for the BERT and LSTM model, respectively. In this paper, we compare the accuracy for each of the training, test, and validation set. Table 1 is the result based on the IMDB dataset. For Table 1, we also compare the "FastText" model from the paper Bag of Tricks for Efficient Text Classification [16] for broader comparisons. Only the highest accuracies achieved were recorded.

	LSTM + Adam	FastText + Adam	BERT + Adam
Training Accuracy	86.98%	86.96%	**92.63%**
Validation Accuracy	86.36%	86.18%	**91.92%**
Test Accuracy	85.28%	85.42%	**91.58%**
Training Loss	0.314	0.435	**0.188**
Validation Loss	0.314	0.363	**0.211**
Test Loss	0.334	0.381	**0.209**
Trainable Parameters	4,810,857	**2,500,301**	112,241,409

Table 1: The BERT model achieves the best performance on sentiment analysis task on the IMDB dataset with 50K movie reviews.

As shown in Table 1, BERT outperforms both LSTM and FastText on sentiment analysis task for larger dataset. Note that the increase in performance comes with a increase in complexity as the BERT model has the most trainable parameters.

	LSTM + Adam	Transformer + Adam
Training Accuracy	**90.64%**	64.28%
Validation Accuracy	**77.92%**	66.74%
Test Accuracy	**75.20%**	66.15%
Training Loss	**0.230**	0.625
Validation Loss	**0.515**	0.605
Test Loss	**0.504**	0.615

Table 2: The LSTM architecture achieves the best performance on sentiment analysis task on the smaller Twitter dataset with 5K tweets.

The results of the models on the smaller Twitter dataset with 5K tweets shows interesting distinction. As shown in Table 2, the LSTM architecture outperforms the BERT model. The results from our experiments show that the BERT model generalizes better on larger dataset but performs poorly on smaller dataset.

6 Conclusion

In this paper we empirically evaluated the performance of bidirectional LSTM and BERT on sentiment analysis tasks for the IMDB dataset containing 50K movie reviews and the Twitter dataset with 5K tweets.

The results clearly demonstrate the superiority of the BERT model on the larger dataset: the accuracy of BERT is significantly higher than the accuracies of the LSTM and FastText models, along with the lowest loss among the three, when performing sentiment analysis task on the 50K IMDB dataset.

There is, however, a trade-off between the sequential operations and decoding complexity. The BERT model has at least twice the number of trainable parameters as the two other architectures and takes much longer to train. The sequential operations in BERT are independent of the sequence length, but they are very expensive to decode [17].

On the other hand, the bidirectional LSTM performs significantly better on the smaller, 5K Twitter dataset and takes much less time to train than the pretrained BERT model.

Unlike LSTM, BERT is extremely scalable with more data and more layers. It allows non sequential parallel computations to reduce training time. In most NLP tasks, including sentiment analysis and speech recognition, the dataset is usually large. Hence, on parallel processing hardwares for longer sequences [17], BERT is still considered a great fit for many NLP tasks.

Different models are suitable for different situations. In this paper, we conclude that the performance of the model is highly dependent on the tasks and the dataset. The results of our experiment can be taken into account when choosing a model for a specific task.

7 Appendix

7.1 Contributions

Equal contribution. Hamin suggested to use the Transformer architecture, experimented model variants, and was involved in implementing the LSTM code. Peng-Cheng suggested to use LSTM and was heavily involved in implementing the LSTM and Transformer code for the experiments.

7.2 Figures

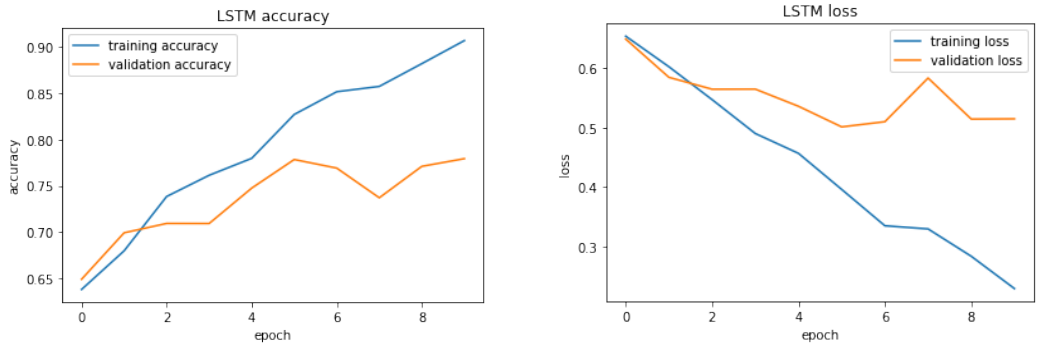

Figure 1: Accuracy and Loss plot for LSTM model on 5K Twitter dataset over 10 epochs

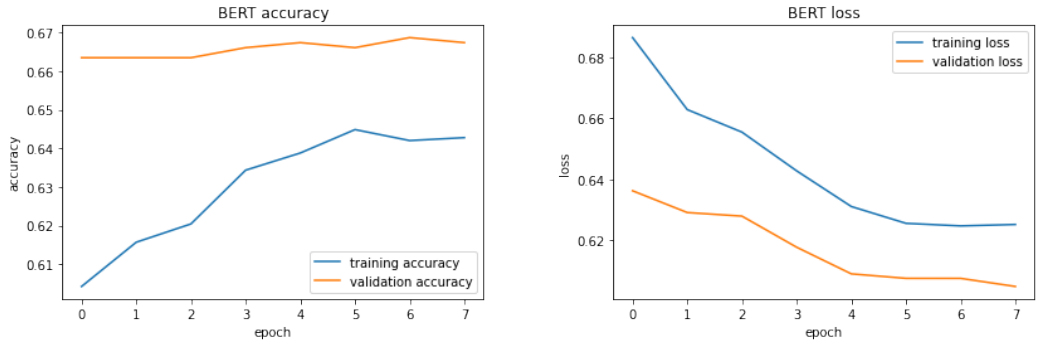

Figure 2: Accuracy and Loss plot for BERT model on 5K Twitter dataset over 8 epochs

References

[1] Hochreiter, Sepp and Schmidhuber, Jurgen. Long short-term memory. Neural computation, 9(8): 1735–1780, 1997.

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

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30, pages 5998–6008, Curran Associates, Inc., 2017. arXiv preprint arXiv:1706.0376

[4] Artaches Ambartsoumian, Fred Popowich. Self-Attention: A Better Building Block for Sentiment Analysis Neural Network Classifiers. arXiv preprint arXiv:1812.07860, 2018

[5] Yoon Kim. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1181. URL http://www.aclweb.org/anthology/D14-1181.

[6] Andrej Karpathy, Justin Johnson, Li Fei-Fei. Visualizing and Understanding Recurrent Networks. arXiv preprint arXiv:1506.02078, 2015

[7] Ben Trevett. PyTorch Sentiment Analysis. URL: `https://github.com/bentrevett/pytorch-sentiment-analysis`. 2021.

[8] Jimmy Ba, Bo Wang, CSC413/2516 Lecture 7: Recurrent Neural Nets and Attention. 2021.

[9] Roger Grosse, Exploding and Vanishing Gradients. 2017.

[10] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Łukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. arXiv preprint arXiv:1801.10198, 2018.

[11] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. Disan: Directional self-attention network for rnn/cnn-free language understanding. In AAAI Conference on Artificial Intelligence, 2018.

[12] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, and Alexander Ku. Image transformer. arXiv preprint arXiv:1802.05751, 2018.

[13] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, Rebecca Passonneau. Sentiment Analysis of Twitter Data. ACL., 2011.

[14] Chaithanya Kumar A. Twitter and Reddit Sentiment Analysis. Kaggle. 2020.

[15] Diederik P. Kingma. Jimmy Lei Ba. ADAM: A Method For Stochastic Optimization arXiv preprint arXiv:arXiv:1412.6980. 2014.

[16] Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov. Bag of Tricks for Efficient Text Classification. arXiv preprint arXiv:1607.01759. 2016.

[17] Roger Grosse, Jimmy Ba. CSC421 Lecture 16: Attention. `https://www.cs.toronto.edu/~rgrosse/courses/csc421_2019/slides/lec16.pdf`