Sentiment Analysis via Deep Learning

Junda An
jan01@email.wm.edu
College of William and Mary
Williamsburg, VA

ABSTRACT

Long short-term memory (LSTM) has been achieved a great success in sequence to sequence learning. Gated recurrent unit (GRU) is also good at natural language processing. Convolutional Neural Network (CNN) has recently recently shown significantly powerful in sentence classification tasks. In this paper, I will implement these three powerful neural networks to sentiment analysis of movie reviews and compare their performances.

KEYWORDS

neural networks, deep learning, LSTM, GRU, CNN, sentiment analysis

ACM Reference Format:
Junda An. 2018. Sentiment Analysis via Deep Learning. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Long short-term memory (LSTM) has been shown to be a great tool in sequence to sequence learning model in natural language processing [2]. Its architecture enables it to memorize information long time ago and to generate outputs based all previous useful information. Since it has shown a great power in sentence translation, I was wondering if it is also good at sentiment analysis. Therefore, I decided to implement LSTM in movie review sentiment analysis, where the data sets are provided by Kaggle [1]. There is another type of recurrent neural network model, gated recurrent unit (GRU), which can memorize information long time ago as well [3]. However, its architecture is slightly different from LSTM and even simpler. According to Occam's Razor rule, the simplest model that fits the data is also the most plausible. Therefore, I wanted to see GRU's performance compared with LSTM. An interesting finding is that the convolutional neural network (CNN), even with one layer, has achieved strong performance on sentence classification [4]. Thus, in this paper, I aim to implement these three different models on the movie review sentiment analysis and compare their performances.

Woodstock '18, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

2 DATA

The data used in this project are all from Kaggle and they are all labeled. The input space is movie reviews from the Rotten Tomatoes dataset. The output space is $\{0, 1, 2, 3, 4\}$, representing the sentiment of movie reviews. 0 - negative; 1 - somewhat negative; 2 - neutral; 3 - somewhat positive; 4 - positive. The target function is to correctly classify a given movie review to a particular sentiment class. The hypothesis set is the set of models that I am going to apply, specifically LSTM, GRU, and CNN. Therefore, it is a supervised classification problem.

2.1 Data Preprocessing

Since the movie review has an unstructure form, I preproceseds the data, removing noise and normalizing the input, in order to make learning easier and more efficient. I first converted each text to tokens and removed punctuation tokens, since punctuation tokens are quite relevant to meaning of the sentence. I also removed stop words, which are the most commonly occurring words that do not contribute any deeper meaning to the sentence. It is worth to notice that stop words include some negation words like "not" and "couldn't", which do change the meaning of the whole sentence. Therefore, I kept this kind of words in the original sentence.

In the normalizing process, I converted all numbers to their word equivalents and converted all letters to lowercase. I also did the word stemming, which is the process of reducing a word to its word stem by removing its prefixes or suffixes. It helps reduce the dimension of inputs without changing the meanings of them. I used Porter Stemmer to stem words since it is one of the most easiest and best-known stemming algorithm.

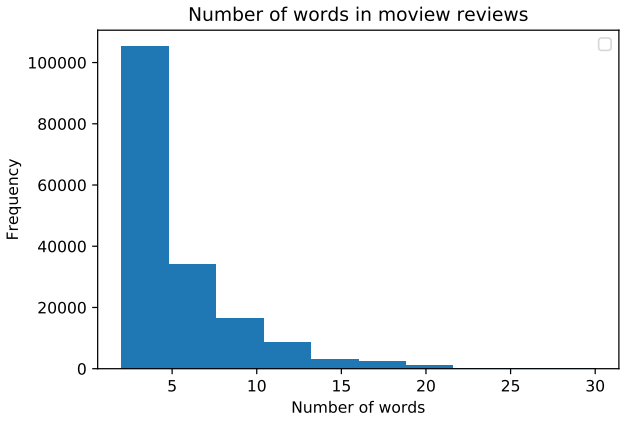

Figure 1: number of words in movie reviews

Since some of movie reviews only contain stopping words, they do not contain any word after I preprocess the data. Thus, I removed these samples. I splited the data set into three parts, namely training set, validation set and testing set. The ratio is $6:2:2$. I have $78,481$ training samples. It is noticeable from Figure 1 that these movie reviews are not long. The maximum number of words among all movie reviews is 30 and the mean is around 5. Therefore, it is sensible to apply GRU in this problem, which has better performance than LSTM on some data sets in which the size of input is relatively small.

2.2 Word Embedding

Word embedding is another important process in natural language processing, in which sequences of words are embedded into high dimensional vectors that contain the sentence meaning. Then, I can feed these vectors to LSTM to do the learning.

The vocabulary in our samples is around 15000 and the maximum number of words in the movie reviews is 30. I embedded word into a vector with dimension 128, so each movie review was embedded into a 30×128 matrix. For LSTM and GRU, this matrix can be regarded as 30 128−dimensional vectors which are feed into the model in sequence. For CNN, this matrix can be directly taken as an input for the convolutional layer.

3 MODELS

3.1 LSTM

It is hard for a traditional feedforward neural network to reason about previous events and generate a good output. However, recurrent neural networks address these issues, by using their internal states, which are their memory. LSTM is a special type of recurrent neural networks. It is composed of a cell state, an input gate, an output gate and a forget gate [6]. These components enable LSTM to memorize all useful information it has received and thus to process sequences of inputs, so LSTM does not have long-term dependency problem, a big problem faced by regular RNN that is unable to memorize events happened long time ago. Therefore, LSTM is a proper model to process movie reviews which, in essence, are sequences of words.

My LSTM model has 2 LSTM layers instead of only one layer, since I want it to learn more information. There are 64 and 32 neurons respectively in the first and second layer. The first layer outputs sequences, because they are inputs to the second LSTM layer. The outputs of the second layer are not sequences, because they are feed to the last output layer, in which they are classified into 5 classes. There are 5 neurons in the output layer, referring to 5 classes of reviews. Softmax function is used as the activation function in the last layer, since it can produce a distribution of reviews and output the most possible class the review might fall in. There are $1,981,989$ parameters in total in this model.

3.2 GRU

GRU is another special kind of recurrent neural networks, which also has gates. However, it has fewer parameters than LSTM, since it lacks of an output gate. GRUs have been shown to have better performance on certain smaller datasets than LSTMs [7]. The maximum length of a single movie review is 30 and most of these movie reviews are short. Therefore, I speculate GRUs might outperform LSTMs in these task.

My GRU model also has 2 GRU layers and 1 output. There are 64 and 32 neurons respectively in the first and second layer. Like LSTM, the output layer has 5 neurons and softmax as its activation function. There are $1,966,533 parameters$ in total in my GRU model.

3.3 CNN

CNNs have achieved a great success in computer vision, because their convolutional layers and max pooling layers can extract important features. Meanwhile, CNNs have been shown successful in sentence classification as well [4]. Their convolutional layers and max pooling layers are able to extract and capture most important features in sentences.

As I mentioned in the previous section, I embeded sentences into matrix representations. The first layer is 1D convolutional layer, in which the filter has the same number of columns as the input matrix, so that the filter can only stride vertically. 64 2×128 filters with stride size of 1 are applied to 30×128 input matrix, with zero-padding. ReLU is used as activation function in this layer, since it is faster and less computationally expensive. Then, 1D max pooling is applied to the output to capture important features and feed the output from max pooling to a dense layer with 128 neurons and ReLU function. Then, the output of this dense is sent to the output layer with 5 neurons and softmax function. There are $1,945,413$ parameters in total in my CNN model.

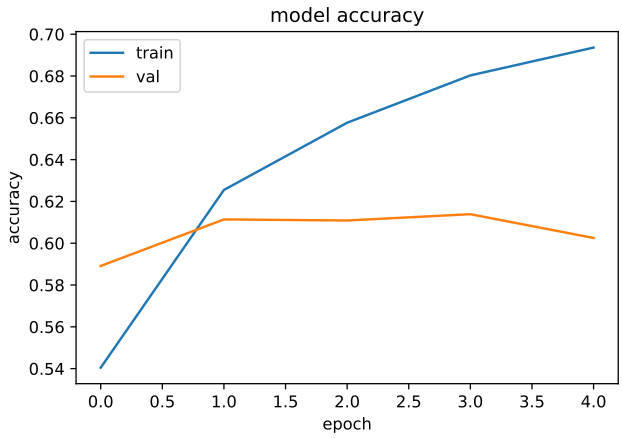

Figure 2: Model accuracy in the CNN model. Validation accuracy drops after the fourth epoch. (0 denotes the first epochs; 1 denotes the second epoch; and so on.)

4 TRAINING DETAILS

4.1 Learning algorithm

I first used stochastic gradient descent (SGD), but the validation accuracy was around 50% after the fifth epoch. Thus, instead of SGD, I used adaptive moment estimation (ADAM) as an optimizer, since it converges much faster than SGD. I used categorical cross entropy as the loss function, since it is a classification problem and

there are five classes. Therefore, the model learns by minimizing the categorical cross entropy loss function.

For all of three models, I set the learning rate to 0.001 and learning rate decay to 1×10^{-5}. I set the learning rate decay, because the model can still learn when the value of loss function is small. I trained my models for a total of 4 epochs. I used batches of 64 samples.

5 REGULARIZATION AND VALIDATION

Even though there are over 78, 000 training samples, parameters in each model are over 1, 900, 00. My models may suffer from overfitting. I applied two regularization methods to overcome overfitting.

First, I used early stopping. I first set number of epochs to 5. I noticed from Figure 2 that the model did not learn much and the validation accuracy even dropped during the last epoch. This is due to overfitting. Therefore, I decreased the epoch number by 1 and the model learned well.

Second, I applied dropout in some layers. For LSTM model, I randomly dropped out 30% neurons in the first LSTM layer, which has 64 neurons, and dropped out 20% neurons in the second LSTM layer, which has 32 neurons. I did the same thing to the GRU model and dropped out the same amount of neurons in each GRU layer as in the LSTM layer. For the CNN model, I dropped out 40% neurons in the dense layer which has 128 neurons.

6 MAIN RESULTS

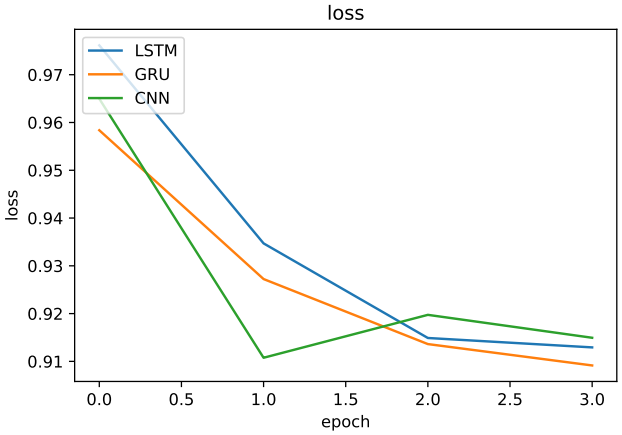

Figure 3: Loss in three models.

From Figure 3, the CNN model converges the fastest, while the LSTM model converges the slowest. It is interesting to observe that the loss of CNN goes up from the second epoch to the third epoch but decreases after the third epoch. This abnormal behavior might be caused by overfitting. Since the CNN model converges rapidly, it achieves the lowest loss at the end of second epoch and starts to overfit afterwards. The figure 4 suggests a similar result. CNN achieves high accuracy first but starts to overfit after the second epoch, even though the accuracy grows back to the highest level at the end of fourth epoch. At the end of fourth epoch, the GRU model achieves the highest accuracy rate, which is 61.45%.

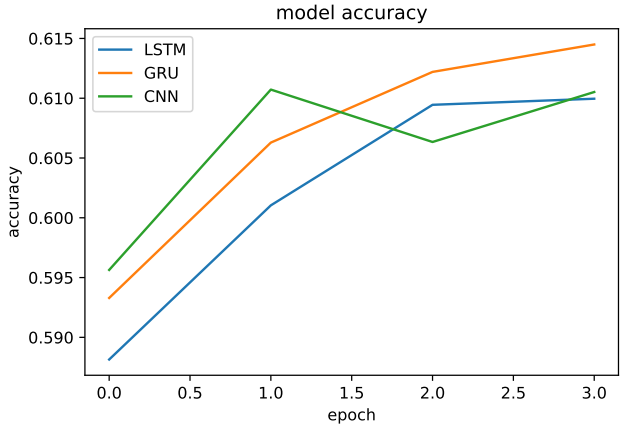

Figure 4: Validation accuracy in three models.

Model	Accuracy on Test Set
LSTM	61.67%
GRU	62.18%
CNN at epoch 2	61.77%
CNN at epoch 4	61.91%

Table 1: Models' accuracy

I applied my model to the test set and got table 1. The GRU model achieves the highest accuracy rate, while LSTM gets the lowest accuracy rate. However, the difference is not very remarkable. These three models all achieve the accuracy rate above 60%, but lower than 70%. Therefore, their abilities to classify movie reviews are pretty much on the same level, while the GRU model is a little better. The loss function and the accuracy on the validation set suggest that CNN might overfit the data after epoch 2, but CNN at epoch 4 achieved 0.18% higher accuracy rate than CNN at epoch 2.

7 RELATED WORKS

Socher [5] built a recursive neural network to do the sentiment analysis. He built the Stanford Sentiment Treebank which is the first corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language. Based on the sentiment treebank, Socher built a recursive neural tensor network (RNTN) which can accurately predict the compositional semantic effects present in the new corpus. A great edge of RNTN over other kinds of neural networks like LSTM and CNN is its ability to analyze semantic compositions. This model achieved the state of the art.

Zhang [7] built a more complicated CNN to classify sentences. The input matrix is divided into 3 regions of sizes 4, 3, 2, each of which has 2 filters. Filters perform convolutions and generate feature maps, over each of which 1-max pooling is applied. These six features are combined together and concatenated into a 2D vector, on which a softmax function is applied. This model achieved the state of art in binary classification. It is worth to notice that Zhang did not increase the depth of CNNs. Instead, he just made

the structure of the CNN a little more complicated, but achieved a better result than [4]. It will be interesting if one can modify Zhang's model to be able to classify sentences into multiple classes.

8 CONCLUSION

In this project, I showed that LSTM, GRU, and CNN can do a decent job in sentiment analysis of moview reviews. They even have potentials to do a better job, because I did not add too many layers to the each neural network and only used a very basic structure. If I modified the structure of these neural networks or simply added more layers to them, I might achieve higher accuracy rate.

Even though the GRU model generates slightly higher accuracy rate, CNN has much more potential, since it can converge quickly. The work by Zhang [7] has achieved higher than 80% accuracy rate in sentence binary classification. One future direction could be to modify the structure of Zhang's CNN to classify sentences into multiple classes.

REFERENCES

[1] 2018. Movie Review Sentiment Analysis (Kernels Only). (2018). https://www.kaggle.com/c/movie-review-sentiment-analysis-kernels-only

[2] Quoc V.Le Illya Sutskever, Oriol Vinyals. 2014. Sequence to Sequence Learning with Neural Networks. *In Advances in Neural Information Processing Systems* (2014).

[3] KyungHyun Cho Yoshua Bengio Junyoung Chung, Caglar Gulcehre. 1979. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *NIPS Deep Learning Workshop* (1979).

[4] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (October 2014).

[5] Jean Y. Wu Jason Chuang Christopher D. Manning Andrew Y. Ng Richard Socher, Alex Perelygin and Christopher Potts. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. *Proceedings of EMNLP* (2013).

[6] Jurgen Schmidhuber Sepp Hochreiter. 1997. LONG SHORT-TERM MEMORY. *Neural Comput.* (1997).

[7] Byron C. Wallace Ye Zhang. 2015. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. *n. arXiv preprint arXiv:1510.03820* (2015).