A Self-attention Based LSTM Network for Text Classification

Ran Jing
School of Control Science and Engineering, Shandong University, Jinan, P.R. China
Email: jingran1997@163.com

Abstract. Neural networks have been used to achieve impressive performance in Natural Language Processing (NLP). Among all algorithms, RNN is a widely used architecture for text classification tasks. The main challenge in sentiment classification is the quantification of the connections between context words in a sentence. Even though various types and structures of model have been proposed, they encounter the problem of gradient vanishing and are unlikely to show the full potential of the network. In this work, we present a new RNN model based on the self-attention mechanism to improve the performance while dealing with long sentences and whole documents. Empirical results show that our model outperforms the state-of-art algorithms.

1. Introduction
Text Classification is one of the most important tasks in NLP. It holds many applications in the research field, such as search engines [1], news analysis [2], information extraction [3], ranking and document classification ([4]; [5]).

Text classification is developing quickly in recent years. In the traditional approach, linear models [6] are adopted on representations of sparse lexical features. Two of these lexical features are bag-of-words (BOW) and n-grams[7]. BOW is based on the frequency of words and hold high computational efficiency, however, the neglect of word order causes information loss. Bag-of-n-gram [8] captures some partial information about word order and improves the performance of BOW. Recently, methods based on deep learning such as CNN [9] and LSTM [10] are widely used by researchers. The advantage of RNN is the strong capacity to capture the extracted features of the context. However, RNN is a biased model (i.e. earlier words can weakly influence the decision when later words appear). Thus, the effectiveness reduces when it deals with whole documents or long sentences. LSTM model proposes to use a LSTM layer to represent each sentence and then passing another RNN variant over these. To some extent, LSTM model can solve the gradient vanishing problem of RNN. It adds a forget gate to keep some information from the previous cell state. Nevertheless, LSTM cannot solve RNN’s problem completely even though it ameliorates the effect of earlier words.

In our work, we set up a self-attention based LSTM model (SATT-LSTM). Our work is closely related to recent works on RNN models for text, such as the work [11] which achieve LSTM application in text classifier and the work [12] which proposed a hierarchical network using attention mechanisms for document classification. The idea of self-attention is similar with certain work [13] which approach the attention model in language translation. Our motivation of this task is to ameliorate the classification’s performance and increase the accuracy while dealing with long sentences.
The remaining of this article is organized as follows. In section 2, the article will introduce the principle of this work in detail. In section 3, our algorithm and framework will be given out. In section 4, the result on all data sets and comparison with other models will be shown. In the end, in section 5, we conclude the paper.

2. Self-attention based text classification model

A common way of the current works is using LSTM model to build the text classifier. LSTM sets up an adaptive gating mechanism. In this mechanism, you can both decide the degree to keep the previous state and maintain current input’s extracted features. Taking a sequence \( S = \{x_1, x_2, ..., x_t\} \) as an example, where \( l \) represents the length of the input sentence, LSTM trains the model word by word. For every time-step \( t \), the hidden state \( h_t \) and the memory cell state \( m_t \) are updated. The algorithm is shown as follows:

\[
\begin{bmatrix}
    \text{In}_t \\
    \text{Fo}_t \\
    \text{Out}_t
\end{bmatrix}
= \begin{bmatrix}
    \mathcal{S} \\
    \mathcal{S} \\
    \mathcal{S}
\end{bmatrix} \cdot W \cdot [h_{t-1}, i_t] 
\]

\[ m_t = \text{Fo}_t \odot m_{t-1} + \text{In}_t \odot \hat{c}_t \]  

\[ h_t = \text{Out}_t \odot T(m_t) \]  

In the equations, \( i_t \) represents the input and \( \hat{c}_t \) shows the present state of the single memory cell. \( \text{In}_t, \text{Fo}_t \) and \( \text{Out}_t \) represent the input gate activate function result, the forget gate activate function result and the output gate activate function result correspondingly. In addition, \( \mathcal{S} \) represents the common logistic sigmoid function we widely used as the activate function and \( T \) represents the \( \text{tanh} \) function.

Even though LSTM is effective in certain situations, there are still a lot of limitations. No matter how long the sentence is, the sentence is encoded as a fixed length hidden vector, so the information of long sentences will be compressed. Thus, the output vector in the last timestep cannot express the meaning of the long sentences accurately. Moreover, the intermediate states’ outputs of LSTM are not fully utilized in this way. An existing way to enhance the performance of LSTM is to add a max pooling layer, but only extracting the maximum value still causes a lot of information loss. To sum up, if we can use every output of each timestep better, we can get a better result. The self-attention model we put forward can cover the drawbacks above-mentioned by using a veritable-length weighted context structure.

The descriptions of our self-attention model are as follows. Given a global context vector of LSTM \( S = \{c_1, c_2, ..., c_n\} \), and \( V \) is the LSTM encoder’s output, \( W \) is the coefficient of linear transformation for \( V \). Thus, we define following statements:

\[ V' = L(V) = W \ast V \]  

Given a variable-length vector \( a_n \) as weights, \( C_w \) is the weighted context. We set a location-based function which combines the current input and the existing sentence statuses by using a veritable-length weighted context as follow:

\[ a_n = \text{softmax}([c_1^T \ast V', c_2^T \ast V', ..., c_j^T \ast V']) \]  

\[ C_w = (c_1 \ast a_n + c_2 \ast a_n + ..., + c_n \ast a_n) \]

\[ \hat{C}_n = [C, C_w] \]  

Then we get the final probability \( p(y|x) = \text{softmax}(\text{MLP} ([\hat{C}_n, h_n])) \) and use the probability to classify the context.
3. The proposed algorithm

The framework of SATT-LSTM is shown in figure 1:

![Figure 1. Algorithmic framework](image)

In SATT-LSTM, we use a LSTM model as the basic model and the unclassified sentence as the input. LSTM encodes the sentence in many timesteps and get the implicit vector for each timestep. The context vector of every timestep is shown in the bottom of the picture with a blue rectangle. And the red rectangle represents the LSTM encoder’s output.

Then we add a global self-attention part on the LSTM model. The dashed part in the picture represents the self-attention framework of SATT-LSTM. $W$ represents the coefficient of linear transformation for $V$. The weight $\alpha_n$ is obtained by passing a softmax layer after doing the inner product between $V'$ and every context vector. The weighted context $C_w$ is the sum of the matrix product of $\alpha_n$ and every context vector $c_j$. Finally, we combine the weighted context $C_w$ and $C$ to get the final prediction.

In this work, the NLL loss [14] is used as the loss function model. In the five-classes classifier, we choose to use the maximum value of the softmax output in the output layer as the prediction result of the text. In the binary classifier, we use a logistic regression layer for the output to get a better result.

4. Experiments and results

4.1. Datasets

In this work, we evaluate the self-attention model on two datasets as follows.

SST-fine: Stanford Sentiment Treebank is a text-emotion dataset which is widely used in text classification. This dataset divides all sentences into 5 categories. The dataset classifies all the movie reviews as fine-grained labels (extremely negative, negative, neutral, positive, extremely positive). This dataset consists of 11840 sentences and are divided into training (8520), validation (1100), and testing (2210) sets.

SST-binary: This dataset divides all sentences into 2 categories. It uses the same data resources as SST-fine. On the other hand, it removes all the neutral reviews and use binary labels of only negative and positive. For the binary classification, the dataset has a split of training (6920) / validation (872) / testing (1821).

4.2. Experimental setup

We implement our model based on Pytorch [15] – an advanced python library, which holds a new transition between eager mode and graph mode to provide both flexibility and speed and supports
GPU acceleration. For datasets, we use accuracy as the evaluation metric. Furthermore, the model’s hyper-parameters are shown in the next paragraph. The dimension of word embedding is 32, and the hidden units of LSTM is 16. For regularization, we employ dropout operation. We tested and finally set the number of LSTM layers as 2. The word vector layer’s and the LSTM layer’s dropout rate are set at the number of 0.5. And we use a fixed learning rate of 0.01.

4.3. Results

As the result, the self-attention model achieves first-class performance on both datasets. We choose several recent researches as the baselines. The details are revealed in table 1:

| Type of network | Model                  | SST-fine | SST-binary |
|-----------------|------------------------|----------|------------|
| ReNN            | DRNN [16]              | 49.8     | 86.6       |
| CNN             | TBCNN [17]             | 51.4     | 87.9       |
| RNN             | RCNN [18]              | 47.21    | -          |
| RNN             | Tree-LSTM [19]         | 51.0     | 88.0       |
| RNN             | BLSTM-2DCNN [20]       | 52.4     | 89.5       |
| RNN             | Multi-Task [21]        | 49.6     | 87.9       |
| Other           | C-LSTM [22]            | 49.2     | 87.8       |
| Our             | SATT-LSTM              | 56.2     | 89.2       |

Generally, the baselines consist of RNN, CNN, LSTM related models and others. CNN model uses a kernel to gather the information, so it is more likely to be influenced by local information. But the relationship between words in a single text and the order of words should be considerate. Even though LSTM model uses the entire sequence as the judgement, the contribution of different parts in the sequence cannot be calculated well. Taking the attention model into account, the contribution of different positions in the sentence can be learnt during the training effectively because the location information is contained by the weighted context. Thus, the self-attention system can increase the accuracy, especially while facing long sentences.

5. Conclusion

This paper introduces a new model in text classifications called SATT-LSTM, which combines the self-attention model and traditional LSTM to improve the ability to handle long sentences in sentiment analysis. SATT-LSTM holds the ability to learn long-term dependencies better by using a veritable-length weighted context in the attention mechanism. We evaluate the learned semantic sentence information on two datasets and make comparisons with related tasks and similar models. In both SST-fine and SST-binary datasets, our model performs well with high accuracy.

In the approaching researches, we will develop more ways to ameliorate the structures of the attention-based model and try to reach better performances.

References

[1] Forman G 2003 An extensive empirical study of feature selection metrics for text classification Journal of machine learning research 3(Mar) pp 1289-1305
[2] Socher R, Perelygin A, Wu J, Chuang J, Manning C D, Ng A and Potts C 2013 Recursive deep models for semantic compositionality over a sentiment treebank Proc. of the 2013 Conf. on empirical methods in natural language processing pp 1631-42
[3] Zeng D, Liu K, Lai S, Zhou G and Zhao J 2014 Relation classification via convolutional deep neural network Proc. of COLING 2014, the 25th Int. Conf. on Computational Linguistics: Technical Papers pp 2335-44
[4] Deerwester S, Dumais S T, Furnas G W, Landauer T K and Harshman R 1990 Indexing by latent semantic analysis Journal of the American society for information science 41(6) 391-407
[5] Pang B and Lee L 2008 Opinion mining and sentiment analysis Foundations and Trends in Information Retrieval 2(1–2) 1-135
[6] Joachims T 1998 Text categorization with support vector machines: Learning with many relevant features European Conf. on machine learning (Springer: Berlin Heidelberg) pp 137-142
[7] Wang S and Manning C D 2012 Baselines and bigrams: Simple, good sentiment and topic classification Proc. of the 50th Annual Meeting of the Association for Computational Linguistics 2 pp 90-94
[8] Joulin A, Grave E, Bojanowski P and Mikolov T 2016 Bag of tricks for efficient text classification arXiv preprint arXiv:1607.01759
[9] Kalchbrenner N, Grefenstette E and Blunsom P 2014 A convolutional neural network for modelling sentences arXiv preprint arXiv:1404.2188
[10] Hochreiter S and Schmidhuber J 1997 Long short-term memory Neural computation 9(8) pp 1735-80
[11] Tang D, Qin B, Feng X and Liu T 2015 Target-dependent sentiment classification with long short term memory arXiv preprint arXiv:1512.01100
[12] Yang Z, Yang D, Dyer C, He X, Smola A and Hovy E 2016 Hierarchical attention networks for document classification Proc. of the 2016 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies pp 1480-89
[13] Luong M T, Pham H and Manning C D 2015 Effective approaches to attention-based neural machine translation arXiv preprint arXiv:1508.04025
[14] Platt J 1999 Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods Advances in large margin classifiers 10(3) 61-74
[15] Paszke A, Gross S, Chintala S and Chanan G 2017 PyTorch
[16] Irsoy O and Cardie C 2014 Deep recursive neural networks for compositionality in language Advances in neural information Proc. systems pp. 2096-104
[17] Mou L, Peng H, Li G, Xu Y, Zhang L and Jin Z 2015 Discriminative neural sentence modeling by tree-based convolution arXiv preprint arXiv:1504.01106
[18] Lai S, Xu L, Liu K and Zhao J 2015 Recurrent Convolutional Neural Networks for Text Classification AAAI 333 pp 2267-73
[19] Tai K S, Socher R and Manning C D 2015 Improved semantic representations from tree-structured long short-term memory networks arXiv preprint arXiv:1503.00075
[20] Zhou P, Qi Z, Zheng S, Xu J, Bao H and Xu B 2016 Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling arXiv preprint arXiv:1611.06639
[21] Liu P, Qiu X and Huang X 2016 Recurrent neural network for text classification with multi-task learning arXiv preprint arXiv:1605.05101
[22] Zhou C, Sun C, Liu Z and Lau F 2015 A C-LSTM neural network for text classification arXiv preprint arXiv:1511.08630