Lijunyi at SemEval-2019 Task 9: An attention-based LSTM model and ensemble of different models for suggestion mining from online reviews and forums

Junyi Li, Haiyan Ding*
School of Information Science and Engineering
Yunnan University, Yunnan, P.R. China
*Corresponding author: hyding@ynu.edu.cn

Abstract

In this paper, we describe a suggestion mining system that participated in SemEval 2019 Task 9, SubTask A - Suggestion Mining from Online Reviews and Forums. Given some suggestions from online reviews and forums that can be classified into suggestion and non-suggestion classes. In this task, we combine the attention mechanism with the LSTM model, which is the final system we submitted. The final submission achieves 14th place in Task 9, SubTask A with the accuracy of 0.6776. After the challenge, we train a series of neural network models such as convolutional neural network(CNN), TextCNN, long short-term memory(LSTM) and C-LSTM. Finally, we make an ensemble on the predictions of these models and get a better result.

1 Introduction

Suggestion mining can be defined as the extraction of suggestions from unstructured text, where the term "suggestions" refers to the expressions of tips, advice, recommendations etc(Negi et al., 2019). These suggestions largely express positive and negative sentiments towards a given entity, but also tend to contain suggestions for improving the entity. Suggestion mining remains a relatively young area compared to Sentiment Analysis, especially in the context of recent advancements in neural network based approaches for learning feature representations. In this task, suggestion mining that classified sentences into suggestion and non-suggestion classes was defined by the organizer.

In this paper, we mainly use an attention-based LSTM model(Hochreiter and Schmidhuber, 1997) for this task. The word-embedding used for all models in this task is Word2Vec. Then, the word vectors are fed into the long short-term memory (LSTM) layer. Finally, an attention mechanism(Luong et al., 2015) is added into the neural networks, and the prediction results are output via the softmax activation. What's more, we try a number of other models (such as the TextCNN(Kim, 2014), the C-LSTM(Zhou et al., 2015) and the attention-based Bi-LSTM(Lai et al., 2015)) for comparative experiments. Furthermore we combine all of the above models to get results by soft voting.

ID: 663_3

Sentence: "Please enable removing language code from the Dev Center "language history" For example if you ever selected "ru" and "ru-ru" laguages and you published this xap to the Store then it causes Tile localization to show the en-us(default) tile localization which is bad."

Label: 1

Figure 1: An example from the SemEval 2019 Task 9 dataset

The rest of our paper is structured as follows. Section 2 introduces models. Section 3 describes data preparation. Experiments and evaluation are described in Section 4. The conclusions are drawn in Section 5.

2 Model

For this task, we use 6 models for experiments. Among these models, the attention-based LSTM models can get the best results. This model combines the attention mechanism with the LSTM. The attention mechanism is a good solution to the information vanish problem in long sequence input situations. When dealing with machine comprehension problems, the LSTM and the attention mechanism are more effective than they are used individually.

For this task, we have 4 chances to submit our result in the final submission. We use differen-

Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019), pages 1208–1212
Minneapolis, Minnesota, USA, June 6–7, 2019.

t methods that are the attention-based LSTM, C-LSTM and ensemble different models.

In this task, we not only select some single models but also use the ensemble model architecture(Sarle, 1996). The ensemble model(Kuncoro et al., 2016) architecture, shown in figure 1, is an ensemble of many single models(We call them sub models)(Dietterich, 2000). Because each sub model is independent of each other, their weights are not shared and just use the same word embedding when training each sub model. The process of the whole ensemble model is carried out model by model. First, each model is run independently, and then the result file is saved. After running all the independent models, the result files are taken out and the final result is determined by the soft vote(Rokach, 2010).

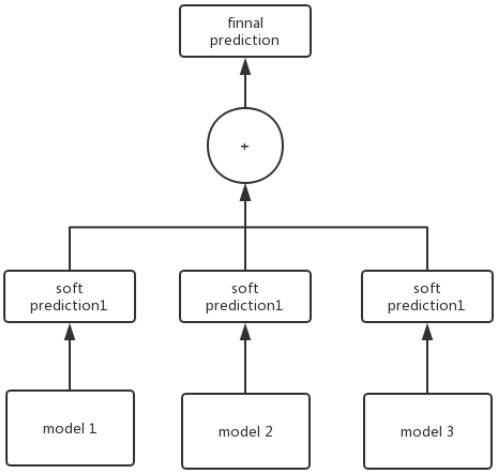

Figure 2: The architecture of the models ensemble

2.1 CNN and TextCNN

The convolutional neural network was originally used to process image data. In recent years, the application of convolutional neural networks has gradually penetrated into many different fields, such as speech recognition and natural language processing. The convolutional neural network consists of three parts. The first part is the input layer. The second part consists of n cyclic layers and collection layers. The third part consists of a fully connected multi-layer perceptual classifier. The difference between a cyclic neural network and a common neural network is that the convolutional neural network consists of a feature extractor with a convolutional layer and a sub-sampling layer. In the convolutional layer, one neuron is only connected to several adjacent neurons.

TextCNN is a model that uses multiple convolutional neural networks to output in tandem (Kim, 2014). In the model, the convolution window of each convolutional neural network is different in size. The convolution results obtained by convolution windows of different sizes are combined and output.

In our task, we also use the basic convolutional neural network and TextCNN to conduct experiments(Zhang and Wallace, 2015). For this task, we find that TextCNN can get a better result than a single convolutional neural network. So, we will be more inclined to choose a TextCNN model instead of a single CNN model for our task.

2.2 LSTM

Traditional recursive neural networks are ineffective when dealing with very long sentences. The LSTM (Hochreiter and Schmidhuber, 1997) model is developed to solve the gradient vanishing or exploding problems in the RNN. Currently, the LSTM is mainly used in natural language processing such as speech recognition and machine translation. Compared with the traditional RNN, an LSTM unit is added to the traditional model for judging the usefulness of information. Each unit mainly contains three gates (the forget gate, the input gate, and the output gate) and a memory cell. The system will judge the usefulness of the information after the input information is fed into an LSTM(Liu et al., 2016). Only the information that matches the rules of the algorithm will be saved, and the other information will be discarded by the forget gate.

2.3 Bi-LSTM

Single direction LSTM(Lai et al., 2015) suffers a weakness of not using the contextual information from the future tokens. Bidirectional LSTM (Bi-LSTM) exploits both the previous and future context by processing the sequence on two directions and generates two independent sequences of LSTM(Kim et al., 2016) output vectors(Liu et al., 2016). One processes the input sequence in the forward direction, while the other processes the input in the backward direction.

In this task, we also use the Bi-LSTM to get a better result(Huang et al., 2015). We select the

model that can be compared with other models as comparative experiments.

2.4 C-LSTM

It has been successfully demonstrated that neural network models can achieve good results in tasks such as sentence and document classification. Convolutional neural networks (CNN) and recurrent neural networks (RNN) are two mainstream methods for this classification task (Zhou et al., 2015). At the same time, these two methods can also be used for our tasks, which use a completely different approach to understanding natural language. In this model, we combine the advantages of both CNN and RNN models and call it C-LSTM for sentence representation and text classification. C-LSTM uses CNN to extract a series of higher-level phrase representations and feeds them to the Long-Term Short-Term Memory Recurrent Neural Network (LSTM) for sentence representation(Stollenga et al., 2015). C-LSTM captures local features of phrases as well as global and temporal sentence semantics. Then, we predict the results based on the labels of the sentences (Zhou et al., 2015).

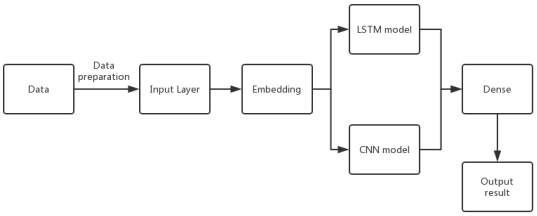

Figure 3: C-LSTM model for our task

In our experiments, the C-LSTM model is compared with a single CNN model, TextCNN, and a single LSTM, Bi-LSTM model. The results show that the C-LSTM model can achieve a better result in this task.

2.5 Attention-based LSTM model

The LSTM model can alleviate the problem of gradient vanishing, but this problem persists in long range reading comprehension contexts. The attention mechanism(Bahdanau et al., 2014) breaks the constraint on fix-length vector as the context vector, and enables the model to focus on those more helpful to outputs. After LSTM layer, we use the attention mechanism on the output vectors produced by previous layer. It is proven effective to improve the performance of our model.

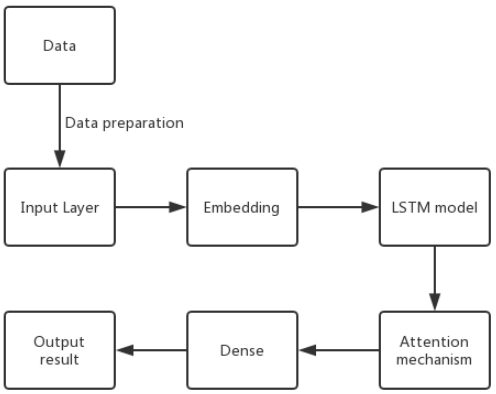

Figure 4: An attention-based LSTM model for our task

In the attention-based LSTM model, all sentences and labels are converted to word vectors by the word embedding layer. These word vectors will be fed to the LSTM layer. Subsequently, the word vector is represented as a hidden vector. Next, the attention mechanism assigns weights to each hidden vector, and the mechanism produces attention weight vectors and weighted hidden representations. Note that the weight vector is mainly obtained by calculating the similarity. An attention weight vector is generated by calculating a sentence vector matrix and a label vector matrix. The attention weight vector is then fed to the softmax layer.

The attention mechanism allows the model to retain some important hidden information when the sentence is long. In our mission, the information of sentences and tags is kept for a relatively long time. Using the standard LSTM may result in the loss of hidden information. To solve this possible problem, we have facilitated the attention-based LSTM model.

In our task, the attention mechanism(Yang et al., 2016) can get better results. We think that the attention mechanism(Vaswani et al., 2017) can improve the efficiency of task. So, we combine the attention mechanism with the LSTM model. This model can get the best results among the single models, which is the final system we submitted.

3 Data Preparation

The organizers provided training, trial, and test sets, containing 8500, 592 and 833 sentences respectively(Negi et al., 2019). Each sentence corresponds to one label, 0 or 1. Although official data is regular, we need to do a further normalization. We want to make it possible to read these sentences easily. First of all, we have completely restored the abbreviated words. For example "i'm not asking microsoft to gives permission like android so any app can take my data" will become "i am not asking microsoft to gives permission like android so any app can take my data". In this sentence "i'm" is an abbreviation. So, we found these abbreviations and restored it by creating a list.

examples	normalization
i'm	i am
doesn't	dose not
can't	can not
i'll	i will
i've	i have
...	...
i'd	i would
it's	it is

Table 1: normalization patterns

Then we noticed that it is also very important to remove some unnecessary characters, such as "!","?" etc. What's more, we find that the link to the web-page is useless for this task. So we remove all urls.

For data pre-processing, we wrote the code to realize the functions and we can improve the efficiency of our final experimental results through these data pre-processing methods.

4 Experiments and evaluation

After data pre-processing, we start the main part of the experiment. The preprocessed data is feed into our prepared model for experimentation. At the same time, we do experiments on different models to compare the test results. In the experiments, we also find that the same model will get different results under different parameter adjustments. For example, we use the C-LSTM model for experiments, and our experimental results range from 0.67 to 0.78 with different parameters in the trial data. Therefore, reasonable adjustment of parameters during the experiment is also a factor in obtaining a good experimental result.

We run each individual model 5 times and use the average as the final result of this model. In all of models, dropout parameters are changed from 0.2 to 0.6, What's more, in the LSTM model, we also select the recurrent dropout (Srivastava et al., 2014) that are set between 0.2 and 0.45. And we set epoch = 10 and batch size = 64.

In this task, we mainly select 6 models and ensemble all of these models. In the table 3 we post the F1-score and recall of the model.

Model	Recall	F1-score
CNN	0.78	0.5523
TextCNN	0.80	0.5908
LSTM	0.81	0.6104
C-LSTM	0.83	0.6222
Attention-BiLSTM	0.85	0.6610
Attention-LSTM	0.84	0.6776
ensemble models	0.82	0.6806

Table 2: Recall and F1-score for each models on task test data

5 Conclusion

In this task, we accomplish this task by integrating LSTM and attention mechanism. After competition, we try various structurally different models and an ensemble of all the models. The performance of a single model is slightly worse than the ensemble model. And there are certain differences between the different parameter results of the same model. Our results are still not as satisfying as the top teams on the leaderboard.

However, in this task, we have some problems which we can't solve. For example, we can not successfully solve the problem of data imbalance. We can not consider the problem of model optimization too much and we don't try more ways of ensemble model.

In the future, we will continue to adjust the model, improve the hardware configuration of the computer, collect more external data, and conduct more experiments to get better results. Furthermore, we will try again to solve the problem of data imbalance. We will continue to do model optimization and we will try more ways of ensemble model.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Thomas G Dietterich. 2000. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer.

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

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, and Noah A Smith. 2016. Distilling an ensemble of greedy dependency parsers into one mst parser. *arXiv preprint arXiv:1609.07561*.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Sapna Negi, Tobias Daudert, and Paul Buitelaar. 2019. Semeval-2019 task 9: Suggestion mining from online reviews and forums. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)*.

Lior Rokach. 2010. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1-2):1–39.

Warren S Sarle. 1996. Stopped training and other remedies for overfitting. *Computing science and statistics*, pages 352–360.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Marijn F Stollenga, Wonmin Byeon, Marcus Liwicki, and Juergen Schmidhuber. 2015. Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation. In *Advances in neural information processing systems*, pages 2998–3006.

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.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Ye Zhang and Byron Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A c-lstm neural network for text classification. *arXiv preprint arXiv:1511.08630*.