An End-to-end solution for Semantic Modeling
CS420 Coursework: Text Classification

Author
joker, 5140309569
Shanghai Jiao Tong University
906476903@qq.com

April 23, 2017

Abstract

The chinese text classification is a pretty important task nowadays. And we study the editor article selection behavior and propose cnn based classification system to automatically select the articles from the large pool. The performance of our model beat the current baseline and reach over 0.905 on the test datasets.

1 Introduction

1.1 Background

Text classification is a very important task in natural language processing. We usually need to classify mountains of the texts and give them true categories. This will cost a lot of time unless we have a proper way to solve that. Researchers have created multiple way, including using the hand-craft feature extract and classify according to keywords, using the bag of words and then logistic regression or SVM classifier. A common feature is that these methods only have a small amount of datasets and only focus on special keywords. This problem happens more on the chinese text classification. And it has a crucial shortcomingnot get full use of the features in the text. So, our method provides a faster and more accurate method, that is to use the convolutional neural networks.

1.2 Formulation

The whole problem can be describe as follows: given a list which has n artices x and n corresponding labels y. The article can be divided into two classes–the positive examples anf negtive examples. And the label can only be 1 or 0. If the label y equals to 1, that means this article is a positive example, otherwise the is a negtive example.

Table 1: Notations and descriptions

Notation	Description
$L(\boldsymbol{x})$	The loss function of the objective $\boldsymbol{x}$.
n-layer conv	a series of layers which will be mentioned later.

And we will construct our model based on these data. And when given a new article, our goal is to predict whether this article is a positive example or a negtive one according to the prediction of our model. And some notations and descriptions are shown in the *Table* 1.

2 Methodology

2.1 Model details

The whole model we used is shown in Fig1.

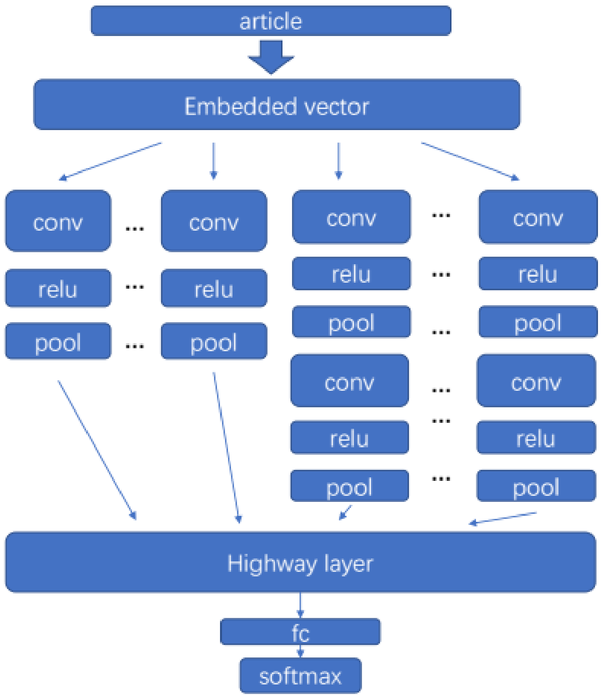

Figure 1: Network

To be specific, when an chinese article is given to the networks, all the words, noted S, will be collected in a dictionary and being encoded to the corresponding label, from 0 to $|S| - 1$. But this can not be the convolutional layers' inputs, because it isn't a suitable representation of the words which have some potential relationship among them.
So, the second step is to do the words embedding. That is to transform every words $c \in S$ into a one dimentional vector v_c. And this is the true input of the convolutional layers.
When it goes to the convolutional layers, we have several layers which have two formats. One is a series of layers contains three layers – the convolutional layer, the relu layer, the pooling layer. The other is twice as large as the previous one, that is the convolutional layer, the relu layer, the pooling layer, then the convolutional layer, the relu layer, the pooling layer. These two can be called one-layer conv and two-layer conv.The we can imagine what the three-layers conv or more. This extracts higher features of the article. And the only

difference of the layers in one format is the different filter size and filter number.
Then, the outputs of all the last pooling layers will be packed and reshaped to meet the need of the next layer – the highway layer. The function of the highway layer can be written as $\lambda * relu(W * x + b) + (1 - \lambda) * x$, where x is the input, W is a matrix, and b is the bias, λ is parameter determined by the input, we note $\lambda = sigmoid(W * x + b)$. The highway can be seen to be a layer that conbine the low level feature with the higher one. And if we just look into details, we will find that it also play a role of kernel function to make the classification easiler.
Finally, the output of the highway layer will be put into a fully connected layer and then through the softmax layer to get the prediction y. Note the output of the fully connected layer is fc. Then

$$y_i = \frac{exp(fc_i)}{\sum\limits_j (fc_j)} \tag{1}$$

The loss function will be (y^* is the one-hot encoded ground truth)

$$L(y) = mean_i(-\sum_j y_j^* log(y_{i,j})) \tag{2}$$

2.2 optimization details

For the optimization method, we use the AdamOptimizer, which has been implemented in Tensorflow. And we set the learning rate to be $2e - 4$, beta1 = 0.9, beta2 = 0.999.

2.3 the advancement

Our model is modified from the networks in [1], what the difference is that instead of using only single-layer convolutional layers, we take the higher level information into account. To be specific, the middle of the networks are consist of several one-layer conv and several two-layer conv. And a highway layer is also added to make the classification be easiler.

2.4 Other thoughts

The model can also be extended to consist of three-layers conv , four-layer convs or more. But due to the limitation of the hardware devices, we cannot try that model. We can also add more models like using the bag of words and SVM and put them to be other streams of the networks, and then get a stronger model.

3 Experiments and Results

3.1 Data preparation

According to the observation of the data, there are far more negetive examples than the postive examples.And this may make the model be more likely to predict the label to be negetive example. Our solution is to just copy the negetive examples eight times to let the number these two kinds of examples almost the same. We also pad the data to the same size.

We choose $\frac{1}{10}$ of the data to be the validation data and the rest to be the train data. The original data are all chinese squences, so we first use the python library *Jieba* to segment the characters to words, and then collected them to a dictionary to get a unique number for each word. So, the input data is these encoded numbers.

3.2 Some Statistics

Give some statistics about the data:

pos examples	neg examples	train examples	val examples	test examples
26702	346090	335513	37279	80478

Table 2: Statistics

3.3 The evaluation metrics and the experimental results

This task is a two-class classification problem, so we use Area under ROC (Receiver operating characteristic) curve to be the evaluation metrics. To be specific,

$$AUC = \frac{\sum_{ins_i \in posclass} rank_{ins_i} - \frac{M*(M+1)}{2}}{M * N} \tag{3}$$

where M is the number of the positive examples, and N is the number of the negetive examples.
During the experiment, we have tried several combinations of these layers in order to achieve higher prformance. And the results are shown in Table 3.

model	AUC of the test data	training time
only a one-layer conv (n-layer conv has been mentioned above) without highway	0.796	about 2h
six one-layer conv without highway choose the best filter size and the filter number we tried	0.869	about 10h
six one-layer conv with highway choose the best filter size and the filter number we tried	0.892	about 10h
six one-layer conv two two-layer conv with highway choose the best filter size and the filter number we tried	0.904	about 12h
six one-layer conv four two-layer conv with highway choose the best filter size and the filter number we tried	0.905	about 12h

Table 3: Statistics

According to the experiment, with the increasing of the number of the one-layer conv, the AUC of the data will increase. But when the number of the one-layer conv increase to six or more, the AUC will not changed a lot. Similarly, we get the number of the two-layer conv to be four. And we cannot try three-layer conv or four-layer conv because of the limitation of the hardware devices.

4 Discussion and Future works

In the experiment, if we choose to use the whole article, the training will be extremely slow. So we only use a part of the article to be the input. But this will cause a problem – if the previous part are the positive example, the rest of the article is negetive, this will make the wrong prediction, although this case is very rare.

When it goes to this task itself, binary classification can only divide the articles into positive or negetive class. In fact, we can make the classification being more accurate, that is to use the k-means or other unsurpervise ways to get the suitable number of the categories, then divide these articles into these classes. This is a more difficult task but more useful.

References

[1] Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.