Project for Machine Learning

Yifan Wang 30
Runxin Liu 40
Hou Xia 30

July 2, 2021

Abstract

Evaluating commentsstars based on comment content is now a hot spot for e-commerce companies. We report on an Evaluation system based on convolutional neural networks (CNN) and an auxiliary classification system based onNaive bayes classifier. The article includes background introduction, two models, experiment process, performance analysis, and a summary.

1 Introduction

Natural language processing is of great significance in various scenarios such as machine translation, keyword retrieval, and human-computer interaction, and is a key concern in the field of machine learning. Star evaluation problem is an important part of natural language processing.

Naive Bayes, as a traditional machine learning method, is a good way to solve the problem of text classification. For example, the classification of spam and non-spam emails. This method is famous for its simplicity but effectiveness. Therefore we choose the Naive Bayes Classifier as one of our models.

Besides, recent years, convolutional neural networks using word vector representation have also made significant achievements in natural language processing. It becomes the current popular machine learning method in NLP. CNN with little hyperparameter tuning and static vectors achieves excellent results on multiple benchmarks [**2**].

What's more, convolutional networks have advantages in extracting the contextual features and emotions of sentences, Which is not available in ordinary neural networks . So we choose CNN as our other model.

In general, considering the excellent performance of Naive Bayes and CNN in NLP, we used the naive Bayes classifier, and a single convolutional layer and single fully connected layer CNN to evaluate stars. And made a comparative analysis of the two.

2 Our two models CNN and Naive Bayes

2.1 Naive Bayes Classifier

Let $|X_i|$ be the number of possible values of the i^{th} attribute of the sentence. As usual, in our naive bayes classifier, we made two assumptions. We assume that the attributes are independent and identically distributed (regarding to the word positions) given their target classification, that is:

$$P(X_i = \omega_k|c) = P(X_j = \omega_k|c), \forall i, j, k, c$$

So to predict the label by naive nayes, we need to compute

$$\hat{y} = \arg\max_{c_i} P(c_i)P(X_1|c_i)\dots P(X_n|n_i)$$

To get $\hat{y}$, there are two kinds of probabilities that we need to calculate:

$$P(c_i) = \frac{\text{the number of i star in all comments}}{\text{the number of comments in the training data}}, i = 1, 2, 3, 4, 5$$

$$P(\omega_k|c_i) = \frac{n_{c_i,k}}{n_{c_i}}$$

Let n_{c_i}be the total number of words in training data whose commentsstars is i, and$n_{c_i,k}$ be the number of times the word ω_i is found among these n_c word positions. Further, we apply the Laplace smoothing technique, to prevent that word ω_i does not appear in the training datas. Thus:

$$P(\omega_k|c_i) = \frac{n_{c_i,k} + 1}{n_i + |V|}$$

Table 1: Naive Bayes classifier Algorithm

Training Naive Bayes Classifier Steps
Input: Training datas after clear and word extraction.
$V \leftarrow$the set of distinct words in the all datas.
textbffor each star c_i in the datas **do**
$V_{c_i} \leftarrow$ the training samples whose star are i
$P(c_i) \leftarrow \frac{\lvert V_{c_i}\rvert}{\lvert V\rvert}$
$n_{c_i} \leftarrow$ the number of word positions in c_i
for each word w_k in vocabulary V **do**
$n_{c_i,k} \leftarrow$ the number of times the word w_k occurs in c_i
$P(\omega_k \lvert c_i) = \frac{n_{c_i,k}+1}{n_i+\lvert V\rvert}$
end for
end for

2.2 Convolutional Neural Networks

Our CNN model is shown in Figure 1.

Build input word vector. To make the input meet the input requirements of the convolutional network, the sentence needs to be converted into a matrix of word vectors. Considering that the one-hot representation method has a lot of redundancy, and the inconvenience of high dimensionality. We use the GloVe that has been trained[**3**]. So input of our CNN are the training data after clear and word extraction.

Build CNN Refer to the paper[**2**], in order to extract contextual connections and emotional information in sentences, we use three sizes of convolution kernels, 3×100, 5×100 and 7× 100 respectively. We need a sufficient number of convolution kernels to ensure the adequacy of feature extraction. So the number of each convolution kernel is set to one hundred. Step size of each convolution kernel is 1.

• Activation function is ReLu. Because from the perspective of calculation, the Sigmoid activation function needs to calculate the exponent, and the complexity High, and ReLU only needs a threshold to get the activation value.

• Let **a** be the output of the convolutional layer, **z** be the input, and w be the convolution weight matrix. σ is ReLu. Forward propagation from input layer to convolutional layer is:

$$\mathbf{a} = \sigma(\mathbf{z}) = \sigma(\mathbf{a} * \mathbf{W} + \mathbf{b})$$

• Maxplooing is used to extract the largest local features and compress the convolution results for redundancy. Forward propagation from the pooling layer to the fully connected

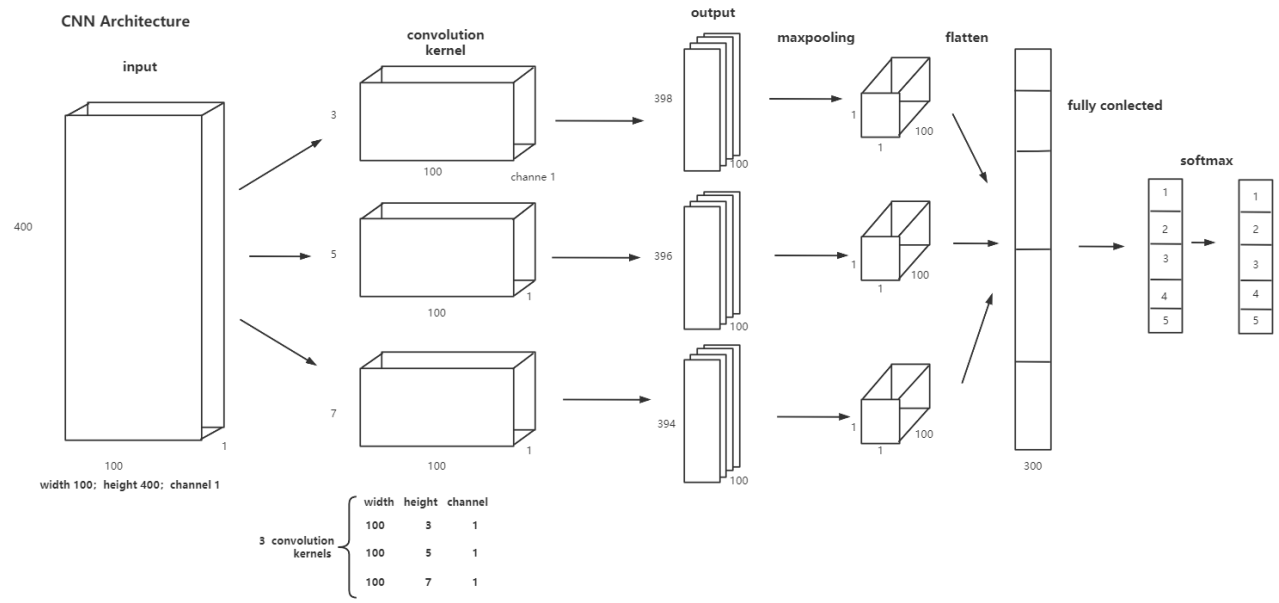

Figure 1: CNN Architecture

layer is:

$$\mathbf{a} = \sigma(\mathbf{z}) = \sigma(\mathbf{W} * \mathbf{a} + \mathbf{b})$$

- Use softmax output,and cross entropy loss function.

$$S_j = \frac{e^{\alpha_j}}{\sum_{j=1}^{T} e^{\alpha_j}}, Loss = -\sum_i Y_i log p_i$$

The backpropagation of the softmax part is relatively complicated.

$$\frac{\partial l}{\partial z} = \frac{\partial l}{\partial p}\frac{\partial p}{\partial z} = \mathbf{p} - \mathbf{y}$$

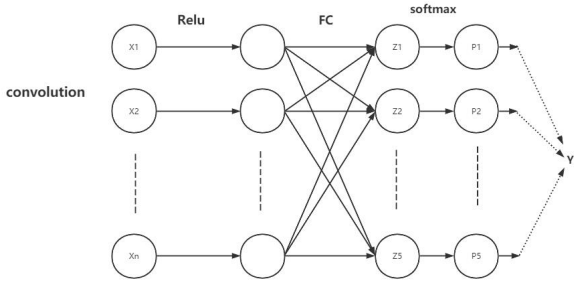

Figure 2: Fully connected layer and softmax

2.3 Experiment

2.3.1 Clean the data

- We first get the CommentsStars and CommentsContent in the csv files. Then, we delete the numbers, Chinese, punctuation, extra spaces and line breaks in the comments and change all words to lowercase. After that, we cut comments into words and remove stop words , like **a**,**the**,**he** and so on, which can improve efficiency and reduce complexity in the later work.
- Secondly, we analyze comments datas and find that we have 83786 comments in total and the average length of one comment is 319 words. The longest one has 12442 words.Therefore, we take the length of each sentence to be 400, and the length of less than 400 to fill in zero.
- Thirdly, we import word vector file globe.6b.100d.txt and generate the word vector matrix for each comment based on this file.

2.3.2 Convolutional network

Divide the data into training set and test set by 8:2. The training set is used to train model parameters, the test machine is used to evaluate model performance, and to prevent model overfitting.

The model parameters are initialized randomly. because if all the parameters are set to the same, the iterative process is only equivalent to one node updating the parameters. Besides random initialization is conducive to finding the optimal solution in an average sense.

Using Mini-Batch Gradient Descent. This method divides the data into several batches, and updates the parameters according to batches. In this way, a group of data in a batch determines the direction of this gradient together, and it is not easy to run off when falling, reducing randomness. On the other hand, because the number of samples in the batch is much smaller than that of the entire data set, the amount of calculation is not very large. It combines the advantages of Batch Gradient Descent and Stochastic Gradient Descent.

One iteration is to use batchsize samples to complete a forward operation and back propagation. Training for all samples is to complete one epoch.We set batchsize=10.

3 Analysis of performance

There are five labels in our experiment, and we divide the five labels into multiple two-label problems for analysis. For example, for a sample with a star of c_1, the correct classification is defined as c_1 as TP, the incorrect classification is defined as not FP, and the sample with a non-star rating of c_1, is defined as FN as classified as c_1 and TN as unclassified as c_1. As for two-label classification problems, there are several performance indicator.

- Accuracy.
$$Accuracy = \frac{Numberofcorrectlyclassifiedsamples}{Totalsamples}$$
This performance evaluation index becomes invalid when the training set sample distribution is very unbalanced.

- Precision.The proportion of samples with positive test results that are actually positive.
$$Precision = \frac{TurePostive}{FlasePostive + TurePostive}$$

- Recall.Proportion of correctly predicted among all positive samples.
$$Recall = \frac{TurePostive}{FlaseNegative + TurePostive}$$

- F1-score. It is the harmonic mean of precision and recall, the maximum is 1, and the minimum is 0.
$$F_1 = 2\frac{Precision * Recall}{Recall + Precision}$$

In our multi-label classification problem, we calculate the above performance indicators for each label, and then take the unweighted average as the overall performance evaluation.

3.1 Naive Bayes Classifier performance

Our naive Bayes classifier has achieved a classification accuracy of 70%. Observing the original data, we find that the five-star sample accounts for about 70% of the total sample, and the data is unevenly distributed.

Since Naive Bayes assumes that the words in the text context are independent, it loses part of the information to a certain extent and is not sensitive to the position of the words. This is the main reason for limiting the performance of Naive Bayes, and it is also the main reason for the simplicity of the Naive Bayes model.

3.2 CNN performance

Table 2: Classification results obtained by CNN

Accuracy	Pricision	Recall	F_1
0.78125	0.19531	0.25	0.21929
0.71875	0.17968	0.25	0.20909
0.8125	0.203125	0.25	0.22414
0.75	0.1875	0.25	0.21428
0.78125	0.19531	0.25	0.21929
0.78125	0.15625	0.2	0.17544

Due to the uneven distribution of the data set, we use $F_1 - score$ as the performance analysis indicator. So far our CNN model has not achieved very good results. Limited by computing resources, our parameter tuning process is slow, but as the number of iterations increases, the model performance can be gradually improved. Analyzing the of the convolutional network, we think that there may be the following points:

- The number of CNN model layers may be insufficient, so it doesn't get a good classification result.
- The trained word vector is directly used, and no adaptive learning is performed on the data set.
- Word extraction, the size and number of convolution windows may all have to be optimized.

Generally, the naive Bayes classifier has achieved good results, but due to its model characteristics, its performance has certain bottlenecks. Convolutional neural networks can theoretically extract contextual information and emotional information in sentences, with proper parameters will have better performance.

References

[1] Tomas Mikolov.. Efficient Estimation of Word Representations in Vector Space.

[2] Yoon Kim. New York University.Convolutional Neural Networks for Sentence Classification.

[3] Jeffrey Pennington. Glove:Global Vectors for Word Representation.