Sentence Sentiment Classification Report

Danrong Li - dl4111

1. **Motivation**

Although it is not a very complicated task to understand if a text is positive sentiment or negative for human-beings, machine learning can make it more convenient. In the case where we need to process a large amount of dataset regarding the opinions for a certain movie, the machine learning algorithm can learn and label these opinions in seconds, whereas human must go through the data piece by piece one-at-a-time. In addition, the human errors are inevitable. Thus, it is crucial to have a machine learning algorithm for sentence sentiment classification.

1. **Original Paper**

The author in the original paper utilized Convolutional Neural Network model. Figure 1 shown below illustrates the model. Suppose we have a sentence with 11 words “a model loading and inference api is now available for scala”, and we represent each word as a 6-dimensional vector. Then we perform 2 different convolution operations with different kernel width to the same input. Then we max-over-time pool them and concatenate the results together. Then we add the fully connected layer with dropout to obtain the final label, whether it is 1 for positive or 0 for negative.

Diagram

Description automatically generated

Fig 1. Text-CNN Model

1. **What I Did**

Firstly, I changed the activation function of convolution network from ‘relu’ to ‘sigmoid’. Since from class, prof Liu mentioned ‘relu’ is largely used in deeper networks to solve vanishing gradients problem, and the text-cnn model is not very deep, I changed the function to ‘sigmoid’. In addition, ‘sigmoid’ function contains the calculation for every z, whereas ‘relu’ function just report the maximum value of z and 0, so I made the change from ‘relu’ to ‘sigmoid’. Figure 2 shown below indicates the performance results of the original ‘relu’ function and the edited ‘sigmoid’ function. According to the figure, the change in the activation function did not make a big impact to the performance. However, the training accuracy for ‘sigmoid’ is higher and the test accuracy gets lower, it might be an indication that ‘sigmoid’ function results in a slight overfit.

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated

Fig 2. Text-cnn model performance results with ‘relu’, ‘sigmoid’ function (from left to right)

Second, I tfidf-vectorized the same dataset, imdb dataset, and then used logistic regression model. Since every entry in the dataframe is not very long, I initially set max\_feature = 100, however, the prediction accuracy is around 0.74. The I set max\_feature = 10,000, and the training accuracy = 0.927 and test accuracy = 0.879.

Third, I used the same tfidf-vectorized dataset, and then used kmeans model. Since the previous models both worked very well, I wonder if the unsupervised cluster can also achieve the good results. I set n\_clusters = 2 and fit the training model, then use the model to predict the test data. The test accuracy is derived from prediction results, and the training accuracy is derived from the model labels after fitting, training accuracy = 0.66, test accuracy = 0.66. From these results, we can see although clustering is better than guessing randomly, it is not as good as logistic regression or the convolution neural network.

1. **References**

Yoon Kim. Convolutional Neural Networks for Sentence Classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP '14).

Google Colab Notebook: <https://colab.research.google.com/github/d2l-ai/d2l-en-colab/blob/master/chapter_natural-language-processing-applications/sentiment-analysis-cnn.ipynb>

Jupyter Notebook: <https://github.com/pytorch/ignite/blob/master/examples/notebooks/TextCNN.ipynb>

1. **My Code**

See next page