COMP9318 Final Project

Group Name: Skilled Driver
Group Members:
z5147182 Xinyuan Qian
z5154763 Yihao Wu

Abstract

This project aims to design an algorithm to fool the binary classifier as much as possible by modifying exactly 20 distinct modifications for each test instance. Our strategy is using linear SVM and using TF-IDF to extra the "best" 20 tokens of each test sample then delete them. Our final success rate is 92%, which means we successfully fool the SVM classifier.

Introduction

The classifier is a binary classifier which allows exactly 20 distinct modifications in each test sample. We are provided a binary classified data to two classes (class-1 and class-0) and a test sample of 200 paragraphs from class-1(test_data.txt).
The aim is to modify the test samples in test_data.txt which belong to class-1 and make them be classified in class-0.

Methodology with justifications

Step1 – Build Bag-of-words model

We decide to use "Bag-of-words" model to represent data. Instead of using implemented package from sk-learn, we implement our function follow three steps:

1. Tokenizing each document
2. Build the token-frequency dictionary
3. Choose token weight

In the token weight choosing step, we decide to use TF-IDF as the weight. Because TF-IDF is a statistical method for assessing in a corpus. It considers not only the frequency of each token, but also the rarity of each token. The importance of a token increases in proportion to its frequency in a file, but meanwhile it decreases inversely with the frequency in the corpus. If a token appears in a class with a high TF and is rarely found in another class, the token is considered to have good class discrimination and is suitable for classification.

Step2 – SVM training

Kernel selection

We tried some different types of kernel, such as polynomial, linear and Gaussian, since the size of vocabulary is 5718 which is larger than the size of training data (540), it turns out the linear kernel is the best choice.

About the other parameters, after several comparisons we found that the influence of these parameters is not important. So we decide to set them as default. Here is the parameters we use.

```
parameters={'kernel':'linear',
            'C':1.0,
            'gamma':'auto',
            'degree':3,
            'coef0':0.0}
```

Step3 – modification-selection

We tried three methods to do the modification. Because the limitation of 20 distinct modifications in each test sample, so we sort the words by their weights.

1. Delete 10 words with maximum weights and add 10 words with minimum weights, the result is 84.5%.
2. Add 20 words with minimum weights, the result is 35%.
3. Delete 20 words with maximum weights, the result is 92%.

Results and Conclusions

modified_data.txt	2018-05-04 03:16:05	Success = 35.000 %
modified_data.txt	2018-05-04 03:27:31	Success = 35.000 %
modified_data.txt	2018-05-04 03:40:10	Success = 72.500 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-04 03:51:04	Success = 84.500 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-04 15:04:39	Success = 74.500 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-04 15:17:39	Success = 95.500 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-04 15:47:21	Success = 95.000 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-04 16:10:38	Success = 95.500 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-04 16:21:58	Success = 92.500 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-04 16:37:25	Success = 92.500 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-05 16:22:45	Success = 92.000 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-05 16:29:59	Success = 92.000 % (Plz make sure that you do not use test data for inference)
modified_data.txt	2018-05-05 16:49:42	Success = 92.000 % (Plz make sure that you do not use test data for inference)

In this project we use bag-of-words model and tf-idf as token weights, and then use linear SVM model to do efficient Text classification. We chose to delete 20 words with maximum weight to fool the classifier and got 92% success.