|  |
| --- |
|  |
| **Newspaper Article Classification Contest** |
| Midway Report |

**Terry LiTao Yu**

*tianweil@andrew.cmu.edu taoyu@andrew.cmu.edu*

**Abstract**

This is the midway report for the Newspaper Article Classification Contest project. The content includes how we retrieve the features from articles, which classifier we used and how to train the classifier. And the performance of our classfier.

**1 Project members**

Terry Li (Andrew ID: tianweil) and Tao Yu (Andrew ID: taoyu)

**2 Feature selection**

**2.1 Feature Extraction and reduction**

Follow the “Bag of words” instruction.

1. Parse the training articles and construct a dictionary for the words.
2. Remove the stop words.
3. Use the word count as the features for classification

After filtering the stop words, we have 38800 words left. Then we use these words to classify the articles.

**3 Train classifier**

For Naïve Bayes classifier,

We use the word count of each to calculate the for every class c.

The equation we used is:

Because a certain word may have zero count in a class, then the is 0. This will cause the posterior probability will become 0. Because there are always some words have 0 count in some classes. It makes all posterior probability for every class is 0, which makes our classifier useless.

To solve this problem, we introduced add-1 smooth to the probability calculation. That is we add 1 count for every feature (different word). To make the total probability of all feature still 1, we need to add the number of features to the total word count.

Then the equation becomes:

For the Prior probability of each class, we just use MLE to calculate them.

After get the and for every word and every class, we finished training the classifier.

**4 Classify Test Data**

For each test document, we also got the word count for each feature. Then

Then we can classify the doc into the class that has the highest

**5 TF-IDF**

To reduce the number of features, we also implemented the tf-idf feature selection algorithm.

The original tf-idf algorithm does not take the class information into account. It only calculates the score of a certain word in a bunch of documents. The original equation is

To introduce the class information into this equation, we used a modified version of tf-idf:

For the tf, the equation is the same:

Because the word count for a certain word in a certain class could be 0, and the doc count that contains a certain word in certain class also could be 0, we introduced add-1 smooth in the process of calculating both tf and idf.

After calculating tf-idf for every feature, we selected 2500 features with the highest scores to be the features of our classifier.

**6 Performance**

|  |  |
| --- | --- |
| Feature selection method | Accuracy |
| 38800 features, no feature reduction | 66.75% |
| 2500 features by tf-idf | 64.8% |

Performance analysis: we think for naïve bayes classifier, the more features the better. So when reduced the feature number to 2500, the performance decreased a little.

But this does not mean our tf-idf is useless. For next step, we plan to try SVM classifier. Then the feature selection is very critical. We are going to try tf-idf and PCA to reduce the number of features.

**References**

[1] <http://en.wikipedia.org/wiki/Latent_Dirichlet_allocation>

[2] <http://nlp.stanford.edu/IR-book/html/htmledition/feature-selection-1.html>

[3] Joachims, Thorsten. Text categorization with support vector machines: Learning with

many relevant features. Springer Berlin Heidelberg, 1998.

[4] Yang, Yiming, and Jan O. Pedersen. "A comparative study on feature selection in text

categorization." ICML. Vol. 97. 1997.