Literary Style Classification with Deep Linguistic Analysis Features

Hyung Jin Kim EVION@STANFORD.EDU
Minjong Chung MJIPEO@STANFORD.EDU
Wonhong Lee WONHONG@STANFORD.EDU

Abstract

Based on the assumption that people in same professional area have similar literacy style, we inferred that similar literary styles can be differentiated by unique terms or expressions they use within the class. In this project, we concentrated on utilizing as many features of various characteristics as possible, and extracting the most meaningful ones from them. As classifier, we used two different types of machine learning algorithms: Support Vector Machine and Naive Bayes Classifier. As a result, our best model successfully classifies the literacy style of authors with 84% accuracy, which is surprisingly better than random guess (33.3%).

1. Introduction

In this project, we demonstrate that selecting deep linguistic analysis features such as semantic relationship frequencies reduce classification error significantly over more commonly used "shallow" features such as frequencies of articles(the-a), pronouns(he-him), pro-sentences(yes,okay).

Our model is built and tested on Twitter tweets. We categorized Twitter's tweets into three different groups: politician group, celebrity group, and technician group. This list of each group is obtained by "wefollow.com", which already classified Twitter users by their profession or interest.

From the data sets, we extract features and implement java classes called feature extractor. Through the feature extractors such as Information Gain Feature Extractor and TF-IDF(Term Frequency-Inverse Document Frequency), we utilize two machine learning algorithms , SVM and NB with the extracted features. Our best result, NB classifier, shows the accuracy of 84% on classification, which is surprisingly better than random guess(33.33%).

2. Prior Works

Before starting to work our project, we research other papers, which are related to our project. We referenced the below three papers.

The first one was "Linguistic correlates of style: authorship classification with deep linguistic analysis features" by Michael Gamon in 2004. Although authorship identification has been a interesting topic in the field of natural language processing, professional identification has not been researched actively. However, with the development of various social network websites, providing professional identification became a big issue to e-commerce companies because the companies can categorize their customers effectively as well as their products. Therefore, we will implement professional identification based on the methods which are used in the first reference paper. The methods of style categorization used on first paper are Frequencies of function words (Monsteller et al. 1964), Word length and sentence length (dating back to 1851 according to Holmes. 1998), Word tags (Aragon et al. 1998), Stability features (Koppel et al. 2003). Based on the above methods, we added several methods for categorizations like: POS tagging, TF-IDF and Manual word selection approach. Details of each methods will be further explained below.

The second one was "Short Text Classification in Twitter to Improve information Filtering" by Bharath Sriram and Dave Fuhry in 2010. This paper classifies incoming tweets into categories such as News(N), Events(E), Opinions (O), and Private Messages (PM), while we classifies the tweets into different categories such as Celebrity group, Politician Group, and Technician Group. Therefore, from our classification model, we can estimate where the writing styles of twitter users are belong to the above three different groups. Also, we use more features such as Term Frequency-Inverse Document Frequency(TFIDF) and POS tagging, and analyze and compare between two different machine learning techniques, Support Vector Machines and Naive Bayes Classifier. Also, because we use the same domain, "Twitter.com", comparing with the reference paper's results can be a meaningful work.

The last one was "Entity Based Sentiment Analysis on Twitter" by Siddharth Batra and Deepak Rao on 2010, which was one of the final project done in the last year's CS224N class. In this paper, the authors worked to extract word clouds for entity words by classifying the opinions as either positive, negative or neutral. The main difference between our project and their approach is that they only consider opinion sentences or phrases. However, sentences or phrases based on fact usually contain more important information, and we try to utilize the fact based tweets in our project.

3. Approach

Our tasks on this project largely concentrated on utilizing as many features of sentences as possible, and reducing the dimensionality of features to avoid over-fitting and to improve the accuracy of the classifiers we built.

3.1. Features

3.1.1. Basic features

Basically, we used the occurrence of each word as a feature (binary feature) by parsing every sentence in training dataset. Since there are lots of irrelevant words in Twitter data such as URL, Twitter ID of users, or typos, we used stem extraction algorithm to find the root word of them and filtered all the inappropriate features. Additionally, we also ignored stop-words as feature words which we believed don't play important role in classification of literary style.

Even though removing seemingly irrelevant words makes sense at first, we found out that sometimes using stopwords or those grammatically incorrect words as features can lead to better performance. It happens since although the word "the" can occur on many sentence, the frequencies or positions of the word in sentence may be meaningful for classification.

Instead of using binary value, we also tried to give a weight to the value of each feature dimension. We used TF-IDF weighting to encode the importance of each feature.

3.1.2. Syntactic features

To incorporate the syntactic information of sentences, we brought the power of the well-stabilized POS tagger. We found out that concatenate POS tag to the word, and using it as feature can lead to the improvement of the performance on some classifiers. Equipped with POS tagger, we were able to select specific class of words for feature such as nouns, verbs, or adjective, some of which shows slight improvement when using solely.

3.1.3. Semantic features

Synonyms of a word can be thought to share certain meaningful semantic information which we cannot extract syntactically. In the end of this project, we could be able to select smaller number of feature words which are more effective than others by using a variety of techniques, as well as the words manually selected by using human intelligence. The intuition we had is that we may be able to use synonyms of the words as feature.

3.1.4. Manual features

We realized that there is certain limitation of performance we can achieve with automatically collected features, and clearly, there are some more useful pattern we can utilize in sentence.

For example, we could easily find out that celebrities tend to use lots of exclamation mark(!) at the end of sentence. Politicians have tendency to mention specific dates frequently as well as some other numeric data like "10% GDP growth". Since these kinds of information are hard to be automatically selected, we manually specified them by carefully looking at the dataset. Some of manual features we engineered are followed:

- Frequencies of punctuation marks and if they are used in a consecutive manner (e.g. awesome!!!!!!!!)
- Number of capitalized words or If the words which are all capitalized are used (e.g. HI EVERYONE)
- If dates or years (e.g. Mar 3, 1884) are mentioned
- If numbers (e.g. 10%) are mentioned
- If emoticons (e.g. :), :D) are included
- If there is retweet(RT) or any reference to other Twitter user (@) (Twitter specific)

3.2. Feature Selection

When we were dealing with a number of features (basic features or synonym expanded features), we went through severe over-fitting problem, especially when we used SVM classifier other than Naive Bayes. It was easy to know this fact since the training error rate was nearly 99% with over 10000 features even though the testing error rate was quite low which was below 44%. In order to address this problem, we introduced several feature selection algorithm.

3.2.1. Term Frequency - Inverse Document Frequency (TF-IDF)

The TF-IDF weighting is usually used as a importance measure of a word in document based on its frequency. As a feature selector, we treated the TF-IDF values of each word as the importance of it as a feature.

$$\begin{aligned} tf_{i,j} &= \frac{n_{i,j}}{\sum_k n_{k,j}} \\ idf_i &= log\frac{|D|}{|d : t_i \in d|} \\ (tf - idf)_{i,j} &= tf_{i,j} \times idf_i \end{aligned}$$

Top 10 words with the highest TF-IDF value
search, thanks, day, today, mesa, tonight, time, christmas, night space, video, photo, week, love, people, guy, monster, aquarius, twitter, tomorrow

3.2.2. Information Gain

First of all, we sort all the feature words based on their information gain value, and select the top words which are over specific threshold we set. Intuitively, information gain is a measure about how we can reduce the uncertainty of classification if we know the value of the feature. Clearly, some words are helpful to determine the label of the data, and some of them are followed: Definition of Information Gain:

$$IG(Y|X) = H(Y) - H(Y|X)$$

In the above equation for information gain, H(Y|X) means the average specific conditional entropy of Y.

$$H(Y|X) = -\sum_{j=1}^{j=i} P(X = x_j)H(Y|X = x_j)$$

Top 10 words with highest information gain
wrap, damon, isn, ridge, fell, relative, habit, t-shirt, popular, touchdown, owned, badge, junk,acceptance, plate, delicious, perfectly, terror, religion, pub

3.2.3. Chi-Square

Chi-square feature selection technique measures the the dependency between the feature and the class. Small chi-square value implies the independence between the term and the class. We have chosen the features with biggest chi-square value from each class.We used the following equation to calculate the chi-square value between the term and the class:

a = number of t occurs within class c
b = number of t occurs in other classes
c = number of training examples in c that doesn't contain term t.
d = number of training examples that's not under class c nor contains term t.
n = the total number of training examples.

$$chi - square(term, class) \tag{1}$$

$$= \frac{n * (ad - cb)^2}{(a + c) * (b + d) * (a + b) * (c + d)} \tag{2}$$

3.3. Classifiers

We use 5-fold cross validation to evaluate the performance of our models.

3.3.1. Support vector machines (SVM)

With the given features above, we firstly tried to classify the tweets based on SVM classifier. Also, we allow to choose various parameters such as type of SVM, kernel type and degree in kernel function. We tested various parameter options to find optimized results.

3.3.2. Naive bayes classifier

Simplest form of Naive Bayes classifier works amazingly well on our classification problem.

3.4. Implementation

In implementation, we used state-of-art libraries for NLP algorithms such POS Tagging, word stemming, and stopwords elimination. We used Stanford POSTagger for POS Tagging, and JWI WordNet package developed by MIT for basic dictionary operations such as word stemming or finding synonyms.

Furthermore, we found out that it took quite amount of time to do some language operations on data such as POS Tagging or word stemming. Since running classifier with different combination of parameters or feature selection algorithms was critical for this project, we devised efficient caching system to improve the productivity, which systematically saves and loads every cacheable entity we use such as FeatureExtractor, Feature, and POSTagger instances.

4. Results

4.1. Performance of Feature Selection

In this project, we mainly used the three different feature extractors, "Naive Extractor", which corresponds the basic features mentioned above, "Information Gain Extractor", and "Chi-Squiare Extractor". Internally, Naive Extractor cuts off the number of features, if any threshold is given, based on the TF-IDF value of words.

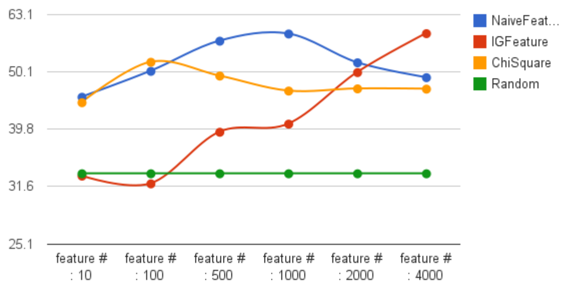

Figure 1. Naive vs Information Gain vs Chi-Square

From the Figure 1, we can obtain the two meaningful points. Firstly, the Naive Feature Extractor which use TF-IDF based feature selection, has the highest classification precision. Secondly, except for Information Gain Feature Extractor, the other two feature extractors shows over-fitting, when they use the certain number of features. The general performance of feature selections and classifications overwhelms the random guesser, which has only 33% accuracy.

4.2. Manual Feature Extractor

From the manual extracting features, we got the important facts that each person in the same categories uses common words or expressions. For instance, politicians use unique words such as "council" and "public", which are related to their professional field. Also, people in the celebrity group frequently use emoticons in their sentences. And this feature greatly contributes to improve our classification model. Without considering use of emoticons, its precision is approximately 7 percents decreased. Another characteristic feature is use of quantitative expressions by politicians. To be more specific, most politicians tend to mention certain dates or days in their tweets to promote their political events.

Figure 2 shows that using manually selected features by human has slightly better performance. In addition, we selected only 35 features, and this fact tells us that only few features are fundamental so that we can estimate people's professions by using small number of features. However, it is not still enough to be a proper classification model to identify people's professions(jobs) by using their writing style.

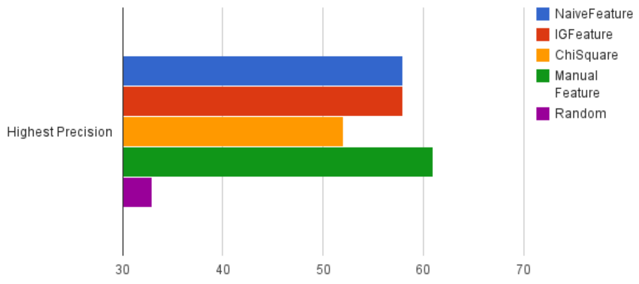

Figure 2. Comparison with Manually Selected Feature Extractor

4.3. Naive Bayes vs Support Vector Machine

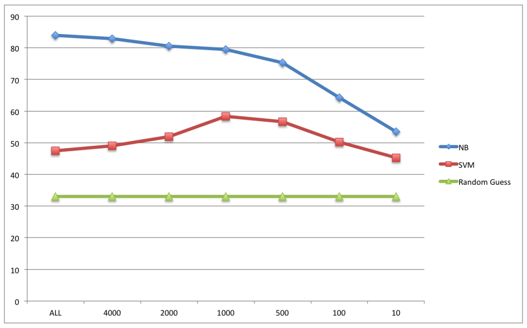

Figure 3. NB vs SVM

From the figure 3, we found some interesting points: Naive Bayes shows the pattern of increased accuracy as number of features increased. Support Vector Machines classifier's accuracy increase as number of features decreases, however after certain point, accuracy decreases. On this figure, we used the naive TF-IDF based feature extraction.

These trends successfully explains characteristics of different types of classifiers. We can understand this with Bias-Variance tradeoff. SVM works poor with many features due to the over-fitting of the data. Selecting only the meaningful features seems to solve this problem(this will be further explained in the feature selection algorithm part), however after certain threshold, precision decreases again due to the lack of number of features. These results matches good with general understanding that NB classifier would work better than SVM under the text classification setting where number of feature is fairly high.

4.4. Syntactic Feature

We used POS tagger to consider syntactic characteristics. From the figure 4, when we used POS tagger, the performance of SVM was fairly improved. After applying POS tagger into our classification model, we wanted to know which class of words such as nouns, verbs, adjectives affects to improve our model.

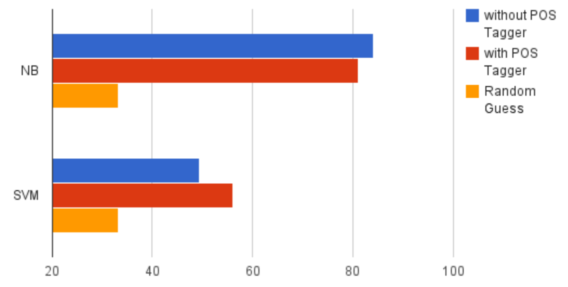

Figure 4. with POS Tagger vs without POS Tagger

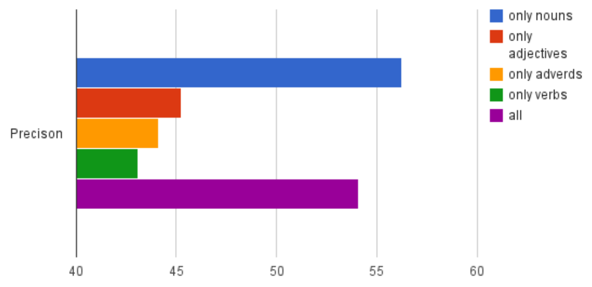

Figure 5. Nouns vs Verbs vs Adjectives vs Adverb vs All

To check which POS tag of words has significant effects on our model, we tested each POS tag of words applied to our model solely. From the test result(figure 5), we found out that nouns's group is a crucial factor to classify twitters' writing style. One interesting point in this experiment is that when we use all POS tag, the performance was slightly decreased because of over-fitting of using many features.

4.5. Semantic Feature

In prior work, we believed that semantic features are strongly related to our learning task. Therefore, we decided to use synonyms of words. The meaning of the max distance in this experiment is the number of used synonyms. For instance, max distance of 0 means that we do not use any synonyms. From the figure 6,

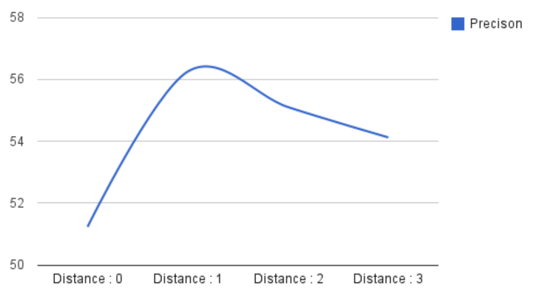

Figure 6. Comparison with Distance of Synonyms(SVM classifier)

we can know that when we used one synonym for each word, the result was dramatically increased. When we use more than one synonyms, its result was almost same or slightly decreased. This decrease of precision performance also can be thought of as the results from over-fitting of using too many features.

5. Discussion

5.1. Dimension Reduction

Our supervised learning problem, which is classifying written texts into several categories, requires various and numerous features. Among the numerous features, finding optimal features is critical to prevent over-fitting because there is only a small number of features that are truly "relevant" to the learning task.

When we use our training set as a test set, we got more than 98% accuracy. In contrast, when we use a distinct test set for testing, the accuracy of our classification model was as low as 45%. This fact means that our machine learning classification model is over-fitted because of a large number of features.

By selecting meaningful features through our feature selection methods, the performance of SVM classifier was drastically improved, which was almost 60% accuracy.

We believe that the performance of our model can be improved by adopting other state-of-art dimension reduction techniques such as Autoencoder, PCA, or ICA.

5.2. POS Tagging and Synonym Expansion

We found out that we could effectively utilize the POS tag information of word in two ways. First, when we used the combination of a word and its POS tag as a feature, the performance was improved on classifiers. For example, instead of using "love", we used "love#N" or "love#V" to represent a feature. Second, we tried to use specific class of POS tag only as features, and we found out that the performance was best when we used only nouns as features. On the other hand, using only verbs had a performance which was almost close to random guess. This result basically indicates that nouns are more relevant and effective to classify the literacy of style of people, especially for deciding the professions of authors.

We also used synonym expansion to utilize the characteristics of appropriate feature words as much as possible. Synonym expansion works based on the assumption that the current feature words are all pretty effective on classification task. We proved that synonym expansion slightly increased the classification accuracy, and we believe that the main reason for the improvement is that it was helpful to handle the data sparsity to the certain extent. Since the NB classifier already handle the data sparsity with proper smoothing techniques, synonym expansion had no effect on the performance of NB classifier.

For future improvement, we believe that incorporating more syntactic information of sentence can be helpful to classify correctly, which include the sentence parsing, dependency parsing, and grammatically correctness of sentence.

5.3. Manually Selected Feature

While testing our model, we printed out a list of frequently used words in correct prediction, and we analyzed the list of the words. From the analysis, we could reach the conclusion that there are some characteristic features such as the length of a sentence, and professional terms to identify each profession group. Because it is hard for machine to train these subtle expressions, we extracted the features manually and compared with other classification models. The results of the model using manually selected features surprisingly worked well, and its result was slightly higher than any other SVM classifiers, which we implemented. However, when we use Naive Bayes classifier, it showed much better performance in terms of classification precision. In our future work, we will try to make a better feature extractor to consider the subtle expressions by human. For instance, we may use clustering algorithms to find some common expressions which the people of same profession use.

5.4. Parameter Optimizations for Classifiers

We realized that the accuracy of SVM classifier largely depends on the combination of various parameters of model. In this project, we set them manually by inspecting the characteristics of the parameters (and also by using the default parameters). We believe that the performance of SVM can be significantly increased by incorporating automatic parameter tuner, which basically tries to find the optimal parameter combination by iterating the set of possible parameter ranges.

6. Conclusion

To solve our problem of classifying various profession from their writing style, we used SVM and NB classifiers with various automatically/manually selected feature sets, and applied several NLP algorithms like POS tagging and word stemming.

As a result, we figured out that Naive Bayes classifier without any additional manipulation yields the best result. It classifies the tweets with accuracy of 84%, which is far superior than the random guess of 33%. This makes sense according to the general agreement about ML classifier: NB works well on the text classification problem, especially on the one with large number of features. This is due to the independence assumption of NB classifier, which reduce the side-effect (over-fitting) of using many features. SVM works relatively poor on our model due to the variance problem of the classifier. Additionally, we also found out a few interesting facts like the one that nouns are the most effective for determining the literary style.

To be most important, we found out that selecting proper features can be critical for better classification of literacy style. Therefore, we believe that this research can be improved by utilizing other dimension reduction techniques such as PCA, Authencoder, which try to automatically extract the most meaningful features.

Furthermore, it is quite difficult to label every data for supervised learning, and most of the time, it should be done by human manually, which is obviously time consuming. We strongly believe that incorporating unsupervised approach such as automatic clustering would lead to more efficient classification.

References

Michael Gamon, *Linguistic correlates of style: authorship classification with deep linguistic analysis features*. 2004.

Bharath Sriram, Dave Fuhry, *Short Text Classification in Twitter to Improve information Filtering*. 2010.

Siddharth Batra, Deepak Rao, *Entity Based Sentiment Analysis on Twitter*. 2010.

Andrew Ng, *CS 229 Machine Learning Lecture Notes3: Support Vector Machines*

http://nlp.stanford.edu/software/tagger.shtml

http://projects.csail.mit.edu/jwi/