Improving Text Classification by Using Conceptual and Contextual Features

Lee S. Jensen
Computer Science Department
Brigham Young University
lee.jensen@whizbang.com

Tony Martinez
Computer Science Department
Brigham Young University
martinez@cs.byu.edu

ABSTRACT

*The exponential growth of text available on the Internet has created a critical need for accurate, fast, and general-purpose text classification algorithms. This paper examines the improvement of broadly based text classification by using features that are easily extracted from training documents. These features represent both the conceptual and contextual properties of a target class, and include synonyms, hypernyms, term frequency, and bigrams of nouns, synonyms and hypernyms. Multiple permutations of the features are applied to three different classification models (Coordinate matching, TF*IDF, and naive Bayes) over three diverse data sets (Reuters, USENET, and folk songs). The findings are also compared to previously published results for the rule-based learner Ripper and results obtained by using another naive Bayes classifier, Rainbow. Suggestions are made about how to automatically determine which features to use, based upon the data set in question. The results demonstrate that the introduction of both conceptual and contextual features decreases the error by as much as 33%.*

INTRODUCTION

Of equal importance to choosing an accurate learning model is the question of what features to use with the model. An accurate model needs to take advantage of the statistical regularity that appears in the training documents. This would include features such as term counts, word sense, and proximity of terms (phrases or collocations). These types of features may be referred to as *contextual* features

Unfortunately, contextual features alone are not enough to achieve the next level of classification accuracy. As several early researchers have noted [1], in order to approach human levels of classification, a learning model must be able to abstract training information and combine it with external knowledge that is not readily available in the training documents themselves. This includes features such as synonyms (words with the same meaning), hypernyms (word hierarchies), and meronyms (part/whole relationships). These kinds of features may be referred to as *conceptual* features, because they take observed terms and group them according to conceptual similarity.

This paper provides a comparison and analysis of 11 different permutations of conceptual and contextual features. Each of the feature permutations is executed using three different learning models. The results demonstrate that the introduction of conceptual and contextual features decreases the error by as much as 33%. Additionally, suggestions are made, based upon empirical results, of which features to use for a given learning model and data set.

Conceptual features

From the possible conceptual feature set, stemming, synonyms, and hypernyms were chosen. Stemming algorithms have broad-based usage in several fields of natural language processing [2]. Two of the more notable uses of synonyms and hypernyms in the area of text classification come from Rodríguez [3] and Scott [4].

Each of these conceptual features assumes knowledge that is not directly available from the training documents. For stems morphological rules for determining roots must be understood independent of the source text. Likewise, synonyms and hypernyms require knowledge of the target language in order to determine their conceptual relationships. In order to provide the external knowledge required by conceptual features WordNet [5] was used.

Contextual Features

Document similarity can often be determined by observing the context of words and phrases used in the documents. A simple approach to representing the contextual relationship of terms is to simply count the number of times a term appears with another term within a given contextual frame. These contextual frequencies can then be used in a similar way as term frequencies. This definition of context is often referred to as bigrams (a context of two terms) or *N*-grams (a context of *N* terms). Much research has focused around bigrams in several different fields of natural language processing [6]. Much of this work reports positive results, especially when working with noun bigrams. Therefore bigrams were chosen as the main contextual feature for this research. Additionally, it logically follows that if a learning model can improve by using bigrams for observed terms, then it should also benefit from the use of bigrams for conceptual features. For this reason bigrams are also calculated for synonyms and hypernyms.

Classification Models

This work uses three different classification models in order to determine the impact of the features discussed above. These models are coordinate matching [7], TF*IDF [7], and naive Bayes [8]. Coordinate matching was chosen because this simple algorithm provides a base that most obviously demonstrates improvements due to the features, and not the power of the classification model itself. TF*IDF was chosen because of its long history of use by the information retrieval community. Naive Bayes is used because it is arguably one of the most accurate current methods for text classification.

EXPERIMENTAL EVALUATION

The accuracy of each feature was determined using three separate data sets: A subset of the Reuters-21578 data; a collection of USENET postings; and a repository of folk songs called the Digital Tradition [www.deltablues.com/folksearch.html].

Coordinate Matching

Table 2 demonstrates the impact of each feature on the coordinate matching classification model. Here we see a base accuracy of 75.68%. The most successful features are bigrams (+5.77%) followed by hypernyms (+3.33%). The combination of these two features creates a total increase of 8.13%, or a total decrease in error of 33.4%.

Feature(s)	*Average*
Base	**75.68%**
Bigram	**81.45%**
Stemming	**74.77%**
Synonym	**77.79%**
Hypernym	**79.01%**
Synonym + Bigram	**82.57%**
Hypernym + Bigram	**83.81%**
Stemming + Bigram	**81.29%**
Stemming + Synonym	**77.00%**
Stemming + Hypernym	**78.04%**
Stem + Hyper + Bigram	**82.35%**

Table 1 – Coordinate matching results

TF*IDF

As was discussed with coordinate matching, bigrams also play a central role in improving classification using the TF*IDF model. As shown in table 3, bigrams improve the base classification of 79.6% by 3.62%. However, in contrast to what was seen with coordinate matching, no feature decreased the accuracy of classification, although the contribution of some features, namely stemming and synonyms, is less then a single percent. The best combination of features includes hypernyms and bigrams and results in an improvement of 3.98% or a decrease in error of 19.5%.

Feature(s)	*Average*
Base	**79.60%**
Bigram	**83.22%**
Stemming	**80.18%**
Synonym	**80.60%**
Hypernym	**81.52%**
Synonym + Bigram	**83.53%**
Hypernym + Bigram	**83.58%**
Stemming + Bigram	**83.18%**
Stemming + Synonym	**80.73%**
Stemming + Hypernym	**80.82%**
Stem + Hyper + Bigram	**81.96%**

*Table 2 – TF*IDF results*

Naive Bayes

Naive Bayes obtained an average base accuracy of 87.44%. With that said, the improvement obtained by using stemming and hypernyms of 1.45%, or 12.5% decrease in error, is significant. With this classification model the most significant feature is stemming (+1.56%), followed closely by hypernyms (+1.14%) and synonyms (+1.06%).

Feature(s)	*Average*
Base	**87.44%**
Bigram	**85.88%**
Stemming	**88.75%**
Synonym	**88.50%**
Hypernym	**88.58%**
Synonym + Bigram	**83.55%**
Hypernym + Bigram	**80.75%**
Stemming + Bigram	**86.23%**
Stemming + Synonym	**88.85%**
Stemming + Hypernym	**88.89%**
Stem + Hyper + Bigram	**84.76%**

Table 3 – Naive Bayes results

CONCLUSION

These findings suggest the need to carefully choose the features and model used to classify a given data set. For this effort automatic sampling and evaluation methods could be used, or perhaps even a more sophisticated supervised/unsupervised learning technique could be employed. However, lacking such automated intelligence the following list of generalities may be consulted when considering what features to implement for classification.

- Always use hypernyms.
- Hypernyms usually improve over the use of synonyms alone.
- Use stemming when the learning model weights comparison classes.
- If a large percentage of terms do not exist in WordNet then favor stemming.
- Bigrams help dramatically, but corresponding individual terms should be removed as a feature.

REFERENCES

[1] M. Lennon, D. Pierce, B. Tarry, and P. Willett. 1981. An Evaluation of Some Conflation Algorithms for Information Retrieval. In *Journal of Information Science*, 3, pp. 177-183.

[2] D. Lewis, R. Schapire, and J. Callan. 1996. Training algorithms for linear text classifiers. In *Proceedings of the ACM SIGIR*.

[3] M. Masterman. 1957. The thesaurus in syntax and semantics. In *Mechanical Translation*, 4:1-2.

[4] G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. 1990. Introduction to WordNet: an on-line lexical database. In *International Journal of Lexicography* 3, No. 4, pp. 234-244.

[5] T. Mitchell. 1997. Machine Learning, McGraw Hill.

[6] M. Rodríguez, J. Gómez-Hidalgo, and B. Agudo. 1997. Using WordNet to complement training information in text categorization. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, Tzigov Chark.

[7] H. Schütze. 1999. Automatic Word Sense Discrimination. In *Computational Linguistics*, Vol. 24, No. 1, pp. 97-124.

[8] I. Witten, A. Moffat, and T. Bell. 1994. Managing Gigabytes: Compressing and Indexing Documents and Images. Van Nostrand Reinhold.