Wikipedia-based Text Classification

Joseph Christian G. Noel

Department of Computer Science
Australian National University

October 2010

Abstract

The Internet and even our personal computers are full of unlabeled text content and we could benefit from tools that automatically organize and tag this text content so that we can quickly access the appropriate content when needed. Supervised learning algorithms for text classification that learn from labeled training examples are already available, however they rely on future data being classified to bear similarity to the training data for good performance and they often require a large amount of labeled training data. One major problem with supervised learning approaches is that they cannot easily generalize to terms that they have not seen before. So over time as new terms (like person names) enter the public lexicon, approaches trained on older data may have trouble generalizing to these new terms, which often bear a lot of informational content for purposes of text classification. However, there are rich knowledge sources like Wikipedia that are frequently updated and which carry rich category information about text terms like person names, organizations, and geographical locations, collectively termed named entities. This report investigates whether a knowledge-based approach to text classification that leverages Wikipedia's knowledge of named entities can outperform various supervised learning algorithms. The results show that the algorithm has some advantages to the current supervised learning algorithms, especially when little training data is available, but also a few weaknesses that need further improvement.

1. Introduction

Being able to effectively organize information is one of the great challenges in machine learning. This is especially becoming more and more important with the increasing amount of data that are continuously being generated. One tool for organizing data is automated text topic classification, using a variety of supervised learning algorithms. They are being used behind the scenes in various applications like spam filters for emails, categorizations of news articles, and automated tagging of web pages to make it easier for search engines find them. All this helps people make better use of all their information and not drown in the large amount of data.

One weakness of supervised learning classification algorithms is that they need a large amount training data to build accurate classification models, and even then when it encounters words that weren't seen in the training data it has no idea how to use these for classification. Usually these words are named entities which are proper nouns like people names, locations, and dates. Examples of named entities are "Bill Gates" and "Microsoft."

Named entities contain a lot of semantic information and being able to use this information may result in improvements in areas where supervised learning classification algorithms are weak. We implement a Wikipedia-based classification algorithm that uses named entities together with a Wikipedia knowledge base to classify Wikipedia and Sydney Morning Herald articles. A comparison and analysis of the results of this classifier with the supervised learning classifiers are then reported.

2. Background

2.1 Supervised Text Classification

2.1.1 Overview

Supervised text classification is a form supervised learning [SR02]. In supervised learning, an algorithm builds a classification model by training on a set of properly labeled training data. It then uses this model to try to classify new data. The accuracy of the algorithm and model relies not just on the quality of the algorithm itself but also on the quality of the training data it had trained on.

If x is a set of real-valued features representing a document's contents and y is the classification of a document, then document D is a set of tuples of x and y, and f is a function that returns the correct classification of the document given the contents. The task of text classification can then be formally defined as:

$$D = \{(x,y)\} \, , \; y = f(x) \, , \; y \in \{0, \, 1, ..., \, k\} \, , \; x \in \Re$$

The goal of a classification algorithm is to learn this f function to be able to classify future documents accurately.

2.1.2 Training and Testing Methodology

A supervised learning algorithm needs two types of datasets. First is the properly labeled training data, which the algorithm trains on and constructs a model out of. The more training data an algorithm can train on and the better the quality of the training data, the better the performance of the classifier will be.

After a model is constructed out of the training data, this model is then tested on a different dataset to gauge its accuracy. It is good practice to make sure that the test data is completely different from the training data to be able to accurately check how well the classifier is able to generalize from the training data it trained on and avoid overfitting. One way to reduce overfitting is to tune the classifier parameters, for example the smooth value in Naïve Bayes and the C penalty factor for Support Vector Machines.

There are various metrics to rate how good classifiers are after testing. One such metric used in this report is *Precision at K*. Because the Wikipedia classifier generates a ranked list of categories and their scores, standard binary classification metrics like precision, recall, and F1 scores do not apply. The metric used has to be able to check not just how often the canonical label is ranked highest in the list of categories, but also how often the canonical label is ranked second highest or third highest. Also, often more than one label is appropriate for an article. For example an article about Microsoft can reasonably be categorized under both "Business" and "Technology" classifications. With *Precision at K* it is possible to rate how accurate the classifier is with respect to more than one label.

Another metric used in this report is the confusion matrix. A confusion matrix presents how often one category is confused with another during classification, and can give hints on the weakness of the classifier and where it is being confused by the dataset.

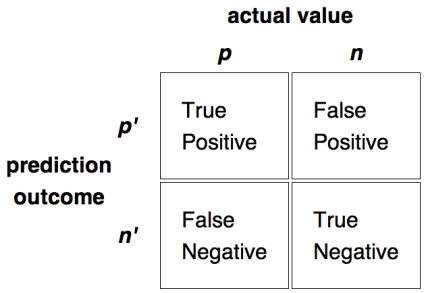

Figure 1: Confusion matrix.
(Retrieved from http://en.wikipedia.org/wiki/Confusion_matrix)

2.1.3 Algorithms

Centroid Classification

Centroid classification is a type of geometric classification [HK00]. Documents are abstracted using a document representation model such as the Term Frequency-Inverse Document Frequency (tf-idf) model. In tf-idf, documents are represented as points in a vector space. Term Frequency is the raw counts of each word in the

document, normalized by the size of the document to prevent bias towards longer documents.

$$tf_{ij} = \frac{n_{ij}}{\sum_k n_{kj}}, \text{ where } n_{ij} \text{ is the count of term } i \text{ in document } j.$$

Inverse Document Frequency is the total number of documents divided by the number of documents containing the term. In this way, words that appear more often in a single document are given greater weight, and words that appear in a fewer number of documents are given greater weight as well.

$$idf_i = \log \frac{|D|}{df_i}$$

$|D|$ is the total number of documents, and df_i is the number of documents that contains term i.

Given the two formulas, the tf-idf for each term i in a document j is then:

$$(tf\text{-}idf)_{ij} = tf_{ij} \times idf_i$$

During training, Centroid gets the vector average of all the documents in a given label to get the centroid vector for that label. These centroid vectors constitute the model of the centroid classifier. To classify a document, it gets the distance of the document's vector model from the centroid vector of each model. The document is then classified under the label of the centroid with the shortest distance from the document vector.

Naïve Bayes Classification

Naïve Bayes (NB) classification is a type of probabilistic classification. There are two types of document representation, the Bernoulli model which abstracts a document as a collection of all the words in the document, and the multinomial model which also takes into account the term frequency of the words in a document [MN98]. The document type used in this report for comparison with the Wikipedia-based classifier is the Bernoulli model. The NB classifier makes use of Bayes' theorem of conditional probability, where the probability of document d being categorized under category i from the set of all categories *C* is:

$$p(C_i \mid d) = \frac{p(C_i)\ p(d \mid C_i)}{\sum_j p(C_j)\ p(d \mid C_j)}$$

The naïve assumption in NB classification is that the probability of a word appearing in a document is independent of the probabilities of the other words in that document, that is, $p(d \mid C_i) = \prod_k p(w_k \mid C_i)$. The final formula for classification is therefore:

$$p(C_i \mid d) = \frac{p(C_i) \prod_k p(w_k \mid c_i)}{\sum_j p(c_j) \prod_k p(w_k \mid c_j)}$$

During training the Naïve Bayes algorithm constructs a model by getting the probability of each word appearing in a document given that the document has a particular label. To classify a document, it just fills in these values into the Bayes' theorem formula to get for each label the probability of being the label of that document. The document is then classified under the label with the highest probability.

One important factor in NB classification is the choice of smoothing value. The smoothing value is a small γ where $0 < \gamma < 1$, that is added to each probability.

$$p(d \mid C_i) = \prod_k (p(w_k \mid C_i) + \gamma)$$

This is because NB multiplies the probability of each word in the document that was seen in the training data for the label. Words that did not appear for a label in the training data will have a probability of 0 and hence makes the whole probability product equal to 0 as well. To work around this, a small non-zero smoothing value is added to each probability so that no words will have a probability of zero. In the experiments for this report, a smooth value of 10^{-12} was found to be the optimal smooth value.

Support Vector Machines

Support Vector Machines are a class of supervised learning classification algorithms that uses a hyperplane separating approach. As in Centroid, documents are modeled as points in a vector space using a document representation model like tf-idf. During training, SVM builds a model by constructing a set of hyperplanes that separates one class of documents from another class with the maximum margin possible. Documents are classified by finding out on which side of a hyperplane they fall under. There are different variations of SVM. The variant used in this report is linear SVM, which performs best for high-dimension vectors such as tf-idf document models. The implementation used in this report is LibSVM which is freely distributed by the National Taiwan University [CC01].

One important variable in linear SVM is the penalty weight for erroneous terms in the training data [CC03]. We call this parameter C and give it a value $C > 0$. Tuning SVM to finding the optimal C is very important for proper classification. The difference between a bad C parameter and a good C parameter on the Wikipedia test data was an improvement from 0% accuracy to 90.63% accuracy when $C=2^{-1}$.

2.2 Named Entity Recognition

Switching tasks now from supervised learning, we next discuss named entities. Named Entity Recognition (NER) is a subset of Natural Language Processing which tries to extract and classify elements in a text into categories such as people names, organizations, dates, currency, and other kinds of proper nouns. For example, given the following text, *"Steve Jobs, the CEO of Apple, Inc., announced the new iPhone 4 on June 7, 2010",* a good NER system should be able to recognize the following words and classifications:

Entity #1: Steve Jobs – Person
Entity #2: Apple, Inc – Organization
Entity #3: iPhone 4 - Object
Entity #4: June 7, 2010 – Date

Named Entities are potentially useful for text classification because of the semantic information that are inherent within them. For example, given an article containing the same text in the example above, it is reasonable to conclude that the article has something to do with technology, without even knowing anything else about the rest of the article.

Additionally, one weakness of supervised learning algorithms is how they deal with words that weren't encountered in the training data. Unencountered named entities are disregarded during classification along with the other unencountered words. If an algorithm can make use of a knowledge base that will allow it to harness the semantic information in named entities, this could potentially increase classification accuracy. This report proposes that Wikipedia can be this knowledge base.

2.3 Wikipedia as a Knowledge Resource

Wikipedia is an online collaborative encyclopedia founded by Jimmy Wales and is run by the Wikimedia foundation. It can be an effective knowledge base resource for named entities because of the sheer amount of data it already contains. It is available in 262 languages, and as of 2010 the English edition already has over 3.4 million articles. Each article consists of a topic which is discussed to varying degrees just like a regular encyclopedia. Each article is classified under a category, and an algorithm can make use of these categorizations to classify documents that contain named entities corresponding to an article.

Categories group together articles on related topics, and are themselves grouped under other categories as well. Categories form a category hierarchy which in most cases approximates a directed acyclic graph. There is a root category, *Contents*, which all other categories are classified below of. And since all articles are classified under at least one category, all articles are classified under *Contents* as well. Figure 1 shows the category hierarchy until a depth of 3 for the category *"Bill Gates"*, which is one of the categories that the article about the Microsoft founder is classified under.

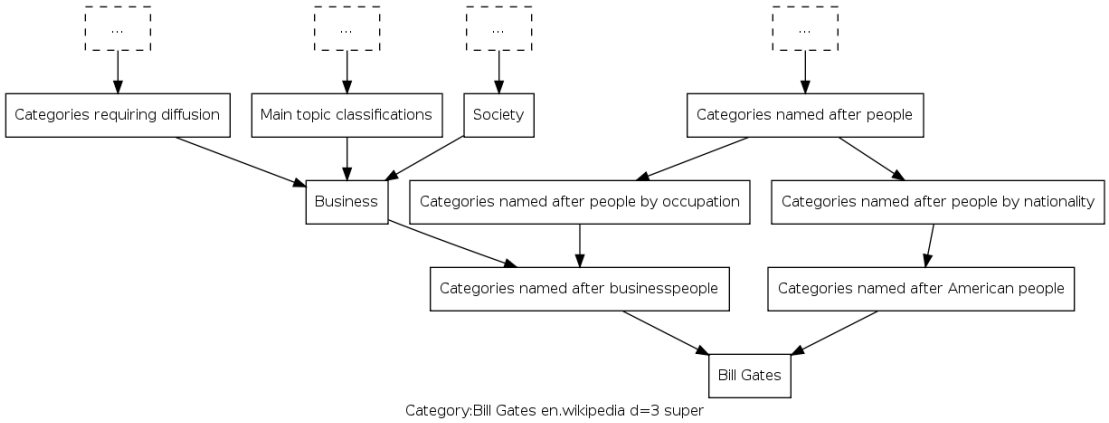

Figure 2: Category hierarchy with a depth of 3 of the category "Bill Gates"

Another feature of Wikipedia that helps in classification of named entities are redirects, which allows different versions of a word to refer to the same article. For example, queries for "Bill Gates" and "William Henry Gates" will both return the same article. This is important because in real world data, the same entity may be referred to in different ways even in the same document. An example of this is in newspapers where the full name of a person is used only the first time the person is referred to, and then only their last names are used to refer to them afterwards.

A major criticism placed upon Wikipedia is that because of its open model of online collaboration where anyone can create and edit an article, it is susceptible to vandalism and unverified or erroneous information. However, a 2005 study by Nature showed that it was close to the level of Encyclopædia Britannica in accuracy and had a similar rate of serious errors [JG05]. Vandalisms are quickly fixed by the community as well.

Also, the semantic information retrieved for a named entity may be ambiguous to the context needed. For example, there is a total of seven "Michael Jordan" articles in Wikipedia, each for a different person with nearly the same name. It is up to the classification algorithm to be able to pass through this ambiguity and get the information it needs, possibly through the categories that the articles are classified under.

3. Wikipedia-based Text Classification

We now turn to the task of designing an algorithm that uses Wikipedia as a knowledge base to harness the semantic information of named entities. The algorithm has the following components:

A Named Entity Recognition system that will extract named entities from the documents that are being classified. For this the classifier is using the NICTA NER, which gives better results than the LingPipe and OpenNLP NER systems.

A library that provides methods for querying Wikipedia for articles corresponding to the named entities. It should be able to handle redirects and disambiguation pages.

The library should also provide methods for getting the article categories, and methods for traversing the category hierarchy as well. The Wikipedia classifier uses the Java Wikipedia Library (JWPL). JWPL uses an offline copy of Wikipedia imported into database tables in MySQL, an open source database.

The basic algorithm for the classifier goes like this. Given a document and a list of possible labels for that document, first extract the named entities in the document. For each named entity in the document get its corresponding Wikipedia article. For each category that the article is classified under, the classifier does a Breadth-First-Search traversal upwards of the category hierarchy. Each time the classifier reaches a category that corresponds to one of the possible labels, that label's score is increased by some value according to a scoring mechanism. The search is stopped after reaching a given depth in the hierarchy, and also after reaching the maximum number of categories considered per entity. This report tries on different values of cutoff depth and maximum number of categories to find the optimal values for both. After all the entities in the document have been processed, the document is then assigned to the label with the highest score among all the possible labels.

One consideration in the Wikipedia classifier is the type of scoring mechanism. A requirement for a good scoring mechanism is that it should penalize categories that are farther up the category hierarchy by giving greater weight to categories that are closer to the entity. The importance of being able to give proper weights to category scores is highlighted by a category hierarchy subset for the article "Bill Gates", shown below. One of the categories the article is classified under is "American Billionaires", and from there a categorization of "Geography" is reached 6 levels higher up the hierarchy. On the other hand, traversing the hierarchy of another category the article is classified under, "Windows People", the category "Computers" can be reached just 3 levels higher. If "Geography" and "Computers" are included in the list of possible labels for the classifier, the category "Computers" should be given a higher score than "Geography" when deciding the classification of an article containing the entity "Bill Gates".

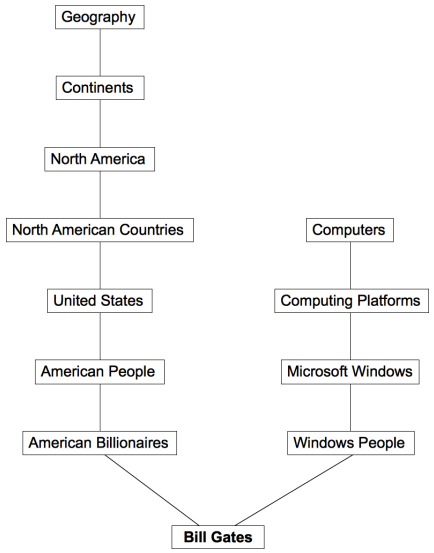

Figure 3: The category path for two categories of the article "Bill Gates"

This report considers two possible scoring mechanisms, the first is where the score of the category is the inverse of its distance, the second is where the score is some value gamma $0 < \gamma < 1$ raised to the power of the distance.

Another thing to consider in the Wikipedia algorithm is the cutoff depth at which to terminate the hierarchy traversal. Cutting off the search at some point reduces noise in the classification scores as categories that are very high in the hierarchy could be erroneous categories already. Also, having a cutoff depth gives the classifier better performance and avoids infinite loops in the hierarchy. Finding the optimum cutoff level is crucial because too high a cutoff and the algorithm encounters too much noise, while too low a cutoff and the algorithm may not be able to reach the useful categories for classification.

The last thing this report considers for the Wikipedia algorithm is the maximum number of categories per entity to include in the scoring. Like the cutoff level, changing the number of categories included has a big impact on the performance of the algorithm. If too many categories are included in the scoring a lot of noise is introduced resulting in too many wrong classifications, and including too few categories may result in the algorithm disregarding correct categories. Also, using the right number of categories per entity may help in disambiguating the context of a named entity through intersections of the parent categories. Tuning both the cutoff depth and the number of categories per entity resulted in big differences in classifier performance. A pseudo-code of the Wikipedia classification algorithm is shown in Algorithm 1.

Wikipedia-based Classifier
Input: Set of documents
Output: Classifications for the input documents

```
foreach document in documents
  possible_categories ← document.possible_categories
  entities ← get_entities(document)

  foreach entity in entities
    found_count ← 0
    article ← get_wikipedia_article(entity)
    category_list ← article.categories

    foreach category in category_list
      if category is in possible_categories and found_count < found_max
        possible_categories[category].score += score(category.distance)
        found_count += 1
      end if

      if category.distance < cutoff
        category_list.add(category.parents)
      endif
    end foreach
  end foreach

  document.label ← max(possible_categories)
endforeach
```

Algorithm 1: The Wikipedia-based classifier

4. Experimental Results

4.1 Datasets

This report uses two sources for datasets. The first dataset comes from Wikipedia and is used for training the base qualifiers and then testing them. For training, a random sampling of 10,000 articles split equally between 10 categories was retrieved from Wikipedia. For testing the classifiers, a random sampling of 1,000 articles split equally between the same 10 categories was retrieved. The dataset is shown in Table 1.

To ensure that the classifiers are able to generalize beyond the Wikipedia dataset it was trained on, they were also tested on articles from the Sydney Morning Herald website. 40 articles from 5 different categories were retrieved from the website, with the categories having unequal number of articles each to better represent real world data. The distribution of the data is shown in Table 2.

Category	No. of articles for training	No. of articles for testing
Culture	1,000	100
Technology	1,000	100
Business	1,000	100
History	1,000	100
Science	1,000	100
Games	1,000	100
Education	1,000	100
Politics	1,000	100
Health	1,000	100
Art	1,000	100

Table 1: Wikipedia dataset distribution

Category	No. of articles for testing
Sports	5
Technology	12
Business	8
Entertainment	4
Environment	11

Table 2: SMH dataset distribution

4.2 Experimental Methodology

Since the supervised learning algorithms make use of testing and training data, a clean separation between the two types of data was enforced so that bias from the training data won't contaminate the testing data.

To generate the test data, 200,000 articles were retrieved from each category. From this 200,000 articles 100 articles were chosen randomly and designated the test data for that category. Because articles are classified under multiple categories and hence may be assigned to "Business" when "History" might be the far better classification, a verification step was added to ensure that the article is being classified under its most logical category according to the Wikipedia category hierarchy. The verification step works simply by traversing up the article's category hierarchy and checking if the category it is being classified under is the closest of the 10 categories to it in the hierarchy.

The training data for the base classifiers are generated with the same process. From the 200,000 articles in each category 1,000 articles are chosen at random and will be the training data for that category. We run each of these articles under the same category verification function to ensure that they are being placed under the correct category. Lastly, one additional check is done to ensure that the article hasn't been already retrieved as test data. This is to ensure that the test and training data stays clean and separate.

For the SMH test data, the author manually got 40 articles from the 5 sections at random in the SMH website.

Tuning is performed on the Wikipedia test data to find the optimal smoothing value for the Naïve Bayes classifier and the C penalty factor for the Support Vector Machine classifier. The same optimal values are used on the SMH test data.

For the Wikipedia classifier, tuning is performed to find the optimum cutoff depth of the category hierarchy traversal and the optimum maximum number of categories considered per entity.

All experiments were performed on a computer with a 2.53 GHz Core 2 Duo CPU and 4 GB of RAM.

4.3 Results

4.3.1 Effect of varying the scoring mechanism of the classifier

First thing to test is how varying the scoring mechanism for the classifier affects the results. This report tests two types of scoring mechanisms. First is the inverse of the distance, the second is some gamma to the power of the distance γ^d. Both the cutoff depth and the number of categories per entities are fixed at 5. The graphs below show the *precision at k* scores for the Wikipedia-based classifier using the different scoring mechanisms.

The results show that the best performance was achieved when $\gamma = 0.5$. The reason γ-scoring works better than the inverse of the distance is because γ-scoring is better able to approximate the actual probability of a category being the correct one.

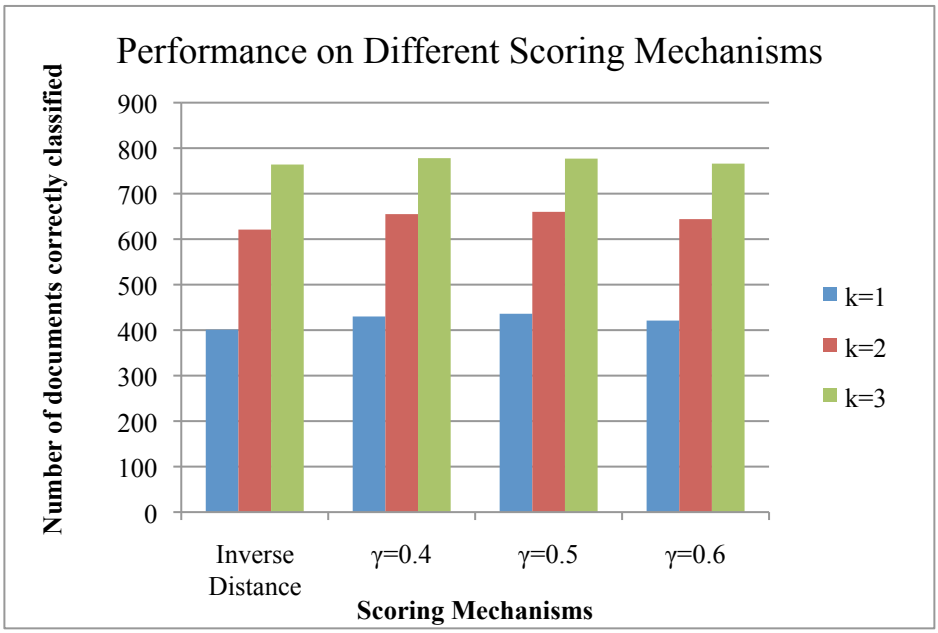

Figure 4: Effect of varying the scoring mechanism

4.3.2 Effect of varying the cutoff level

The second thing to test is the effects of different cutoff depths to the classifier results. Since the best results in the previous section was with γ=0.5, that is the scoring mechanism uses here. As in the previous section, the number of categories included per entity is set at 5.

The results show that the optimum cutoff depth is 4, and generally lower cutoff depths mean better performance. However at cutoff depth lower than 4, the algorithm isn't reaching the useful categories for classification anymore. So apparently to combat against noise in the category hierarchy, the best way is to reduce the cutoff depth in order to reduce the number of categories being reached, rather than trying to drown out the noise by increasing the cutoff depth to include more categories.

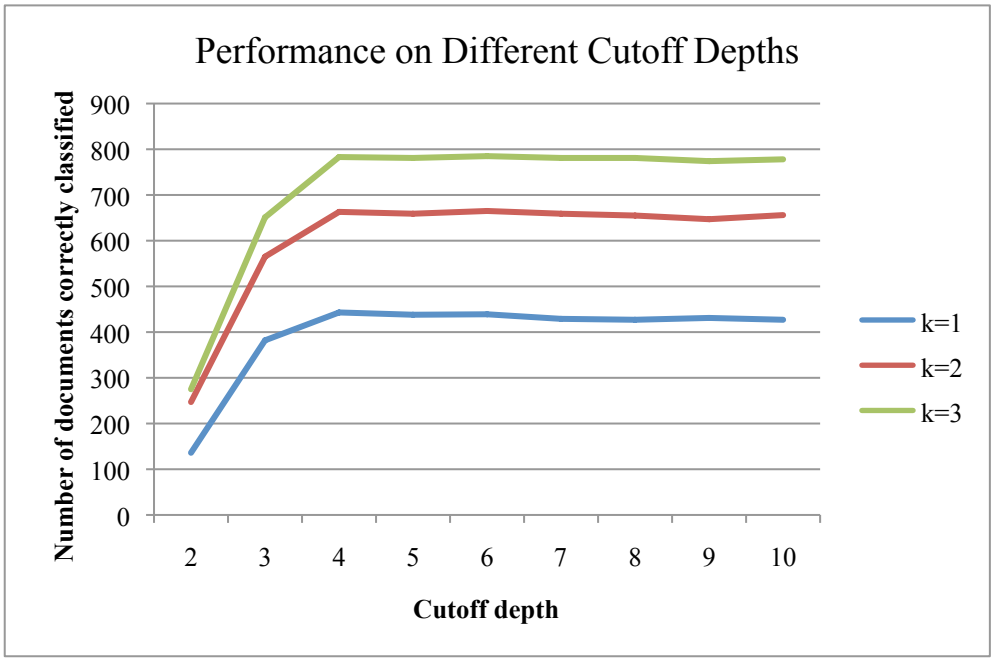

Figure 5: Effects of varying the cutoff depth

4.3.3 Effect of varying the number of considered categories per entity

The third aspect tested is the effects of different values for the max number of considered categories per entity. The same γ=0.5 scoring mechanism as in the previous section is used, and the optimal cutoff depth of 4 is used for all tests in this section.

Reducing the maximum number of considered categories per entity results in better performance by the classifier. In fact, the optimal number was found to be just 1. The results in the last two sections suggest that going too far in the category hierarchy puts too much noise in the category scores, and it would be better to just stick with the first of the possible labels that are found traversing the hierarchy. This can be considered its canonical category.

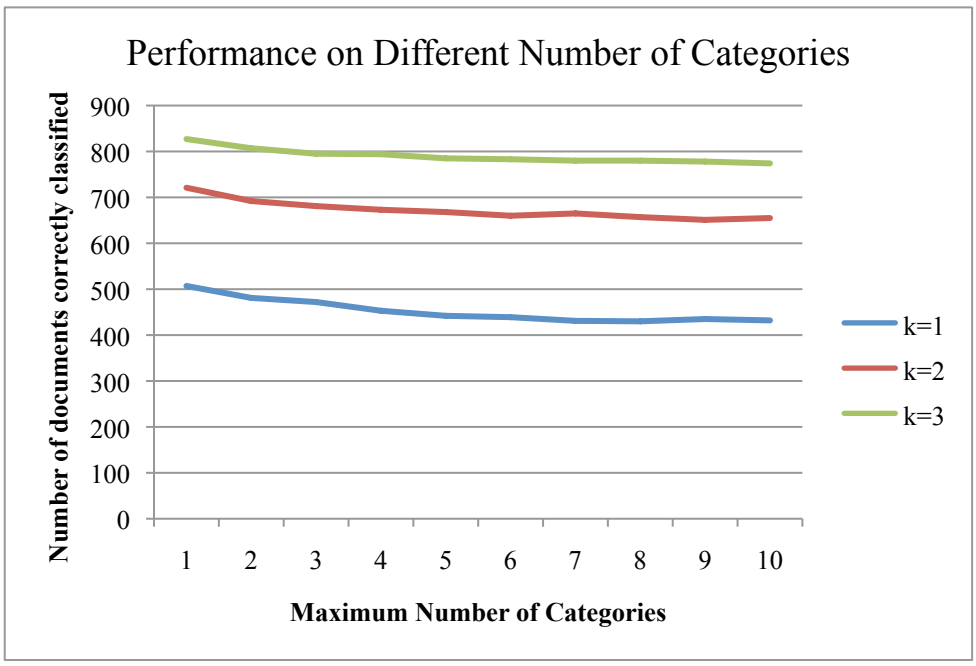

Figure 6: Effects of varying the maximum number of categories per entity

4.3.4 Results of base classifiers

Given a large enough dataset to train on, it was found that the supervised learning classifiers were able to outperform the Wikipedia-based classifier. However once the amount of training data was reduced, there was a corresponding reduction in the performance of the supervised classifiers. Once the number of articles in the training data reached just 100, the Wikipedia-based classifier has already outperformed them.

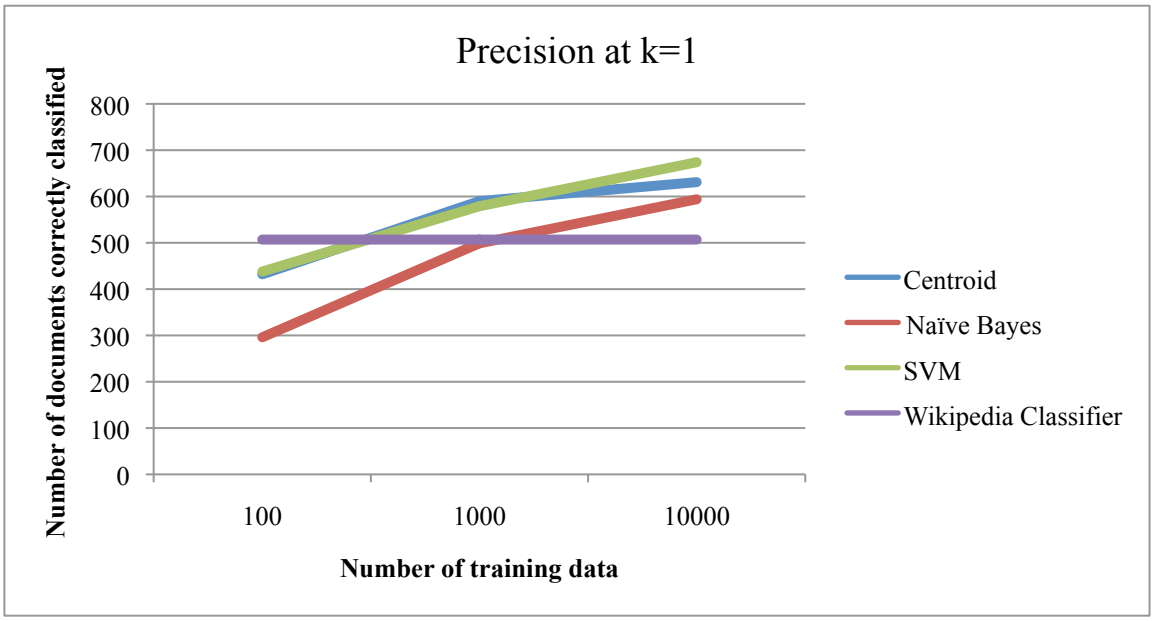

Figure 7: Results of the supervised learning classifiers in comparison with the Wikipedia classifier

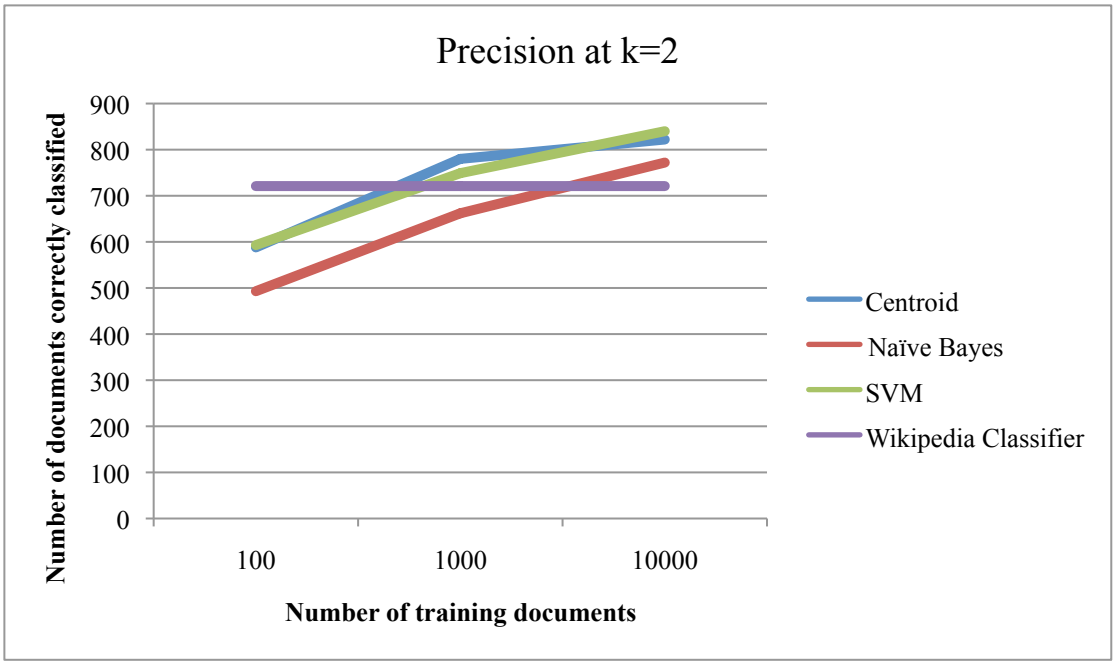

Figure 8: Results of the supervised learning classifiers in comparison with the Wikipedia classifier

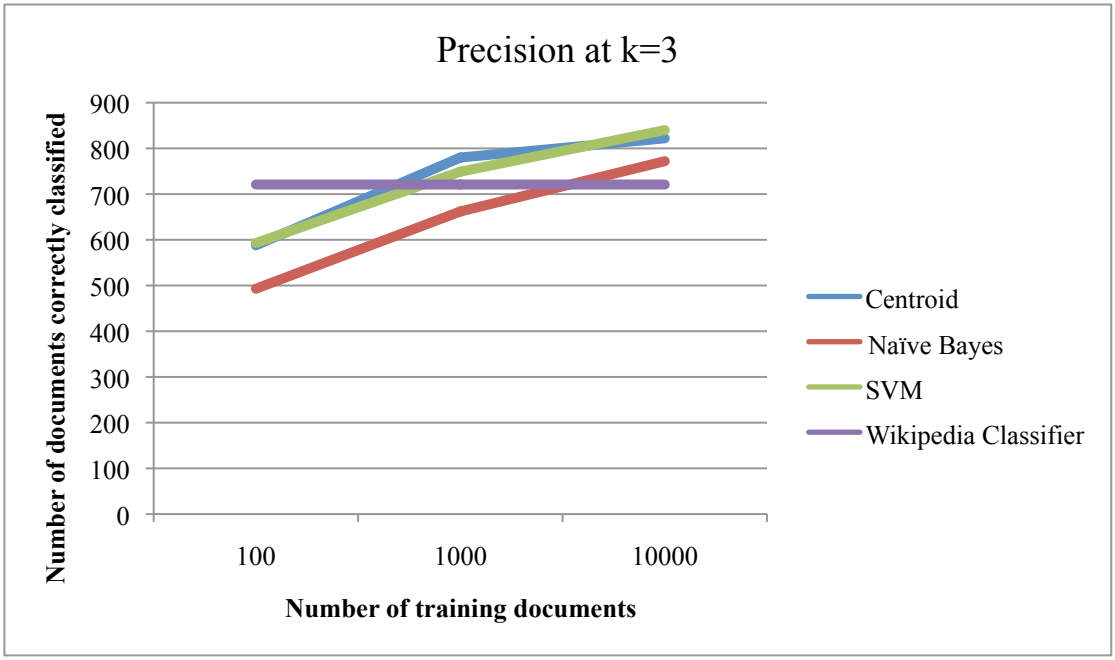

Figure 9: Results of the supervised learning classifiers in comparison with the Wikipedia classifier

4.3.5 Results on SMH data

The Wikipedia-based classifier was able to generalize it's model well, getting almost the same percentage of articles correct at k=1. As in the testing on the Wikipedia test data, the supervised classifiers beat the performance of the Wikipedia-based classifier when training on 10,000 articles, but their performance gradually declined when trained with fewer training data.

After training on 10,000 articles, the supervised learning classifiers already had a feature set of 294,674 unique words. This large feature count reduced the supposed advantage of the Wikipedia-based classifier on unseen words, simply because at that point very few unseen words were seen in the test data anymore. Even when the training data was reduced to 1,000 and then to 100 articles, the feature counts were still 74,191 words and 20,590 words, respectively.

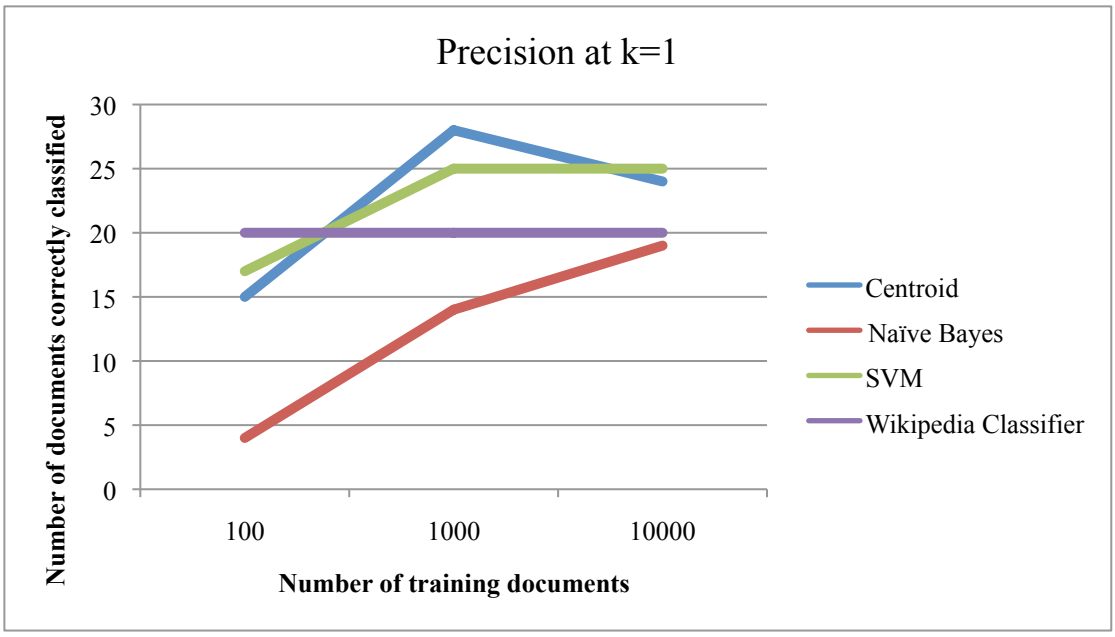

Figure 10: Results on the SMH test data.

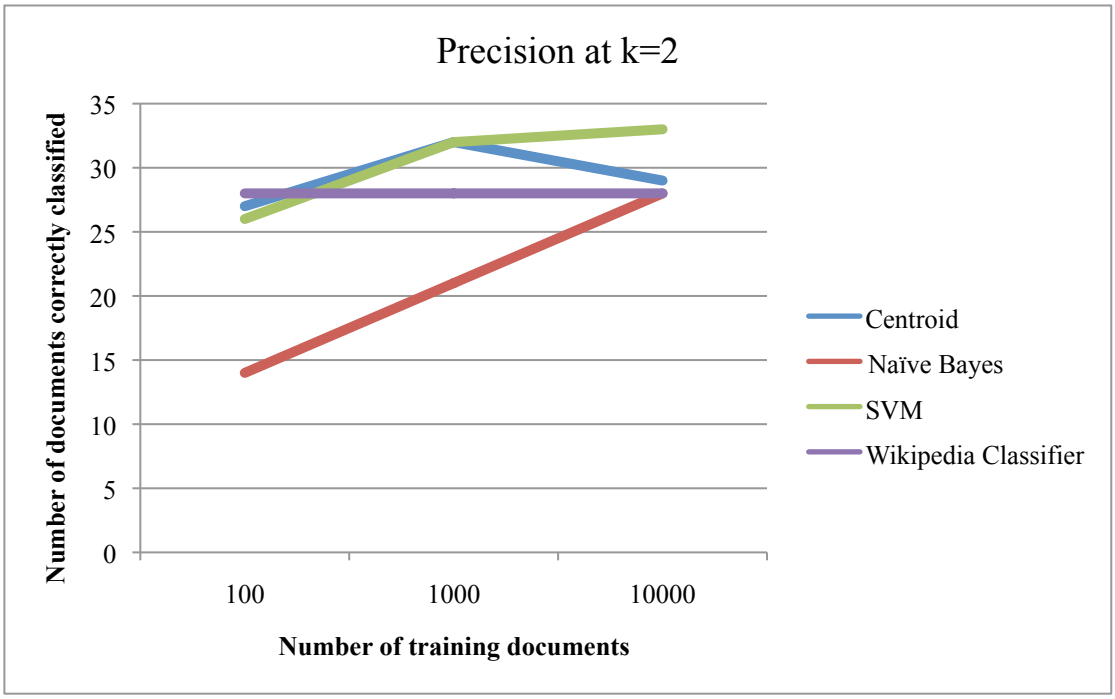

Figure 11: Results on the SMH test data.

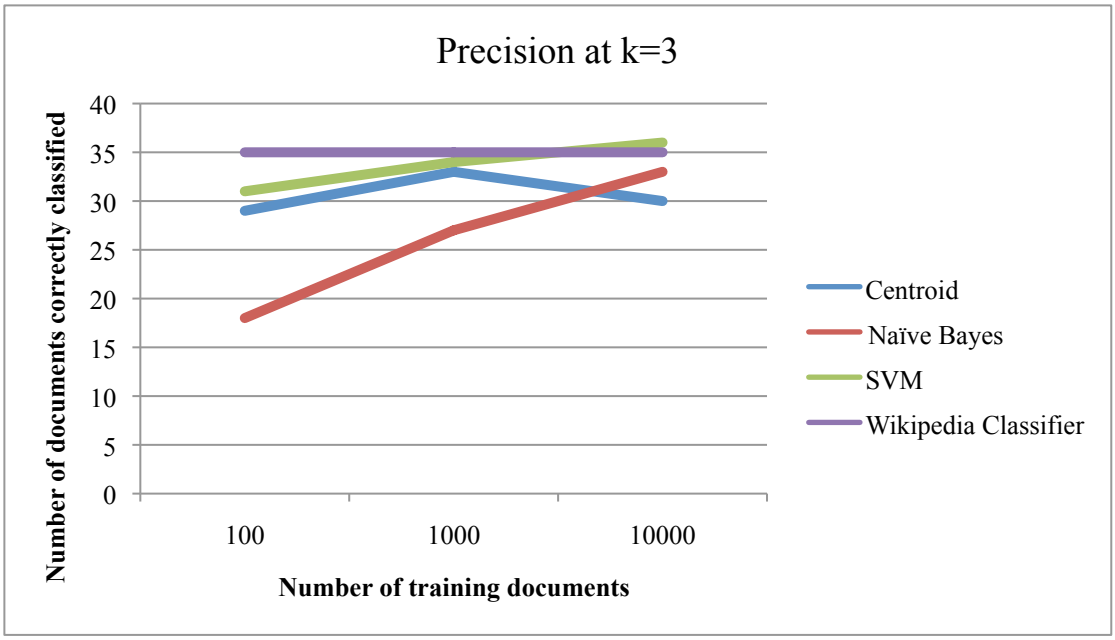

Figure 12: Results on the SMH test data.

4.3.6 Analysis

One problem that the Wikipedia-based classifier had with the SMH data was that it was confusing a lot of articles that were classified under Technology and Environment in SMH as Business articles. Upon investigation it was found that this was because for Wikipedia articles about technology companies like Apple, Microsoft and Motorola, the category "Business" was closer to the article in the category hierarchy than the category "Technology." The same thing happens for articles about companies related to the Environment news. And since most news articles had something to do with technology or environmental companies, the Wikipedia-based classifier was classifying these as Business. Errors like these are understandable, as it results from the different biases in the hierarchy of the training data and the test data.

On the other hand, the Wikipedia-based classifier performed very well in Business, Sports, and Entertainment. Entities found in the articles here had none of the confusion in the category hierarchy that were found in the others.

	Environment	**Business**	**Entertainment**	**Sports**	**Technology**
Environment	1	0	0	0	0
Business	7	7	1	0	8
Entertainment	0	0	3	0	1
Sports	0	0	0	5	0
Technology	1	0	0	0	3

Table 3: Confusion matrix for SMH data.
Vertical columns represent the true classifications. Horizontal rows represent the predicted classifications.

We now take a look at a specific case where the supervised learning classifiers were able to beat the Wikipedia-based classifier and vice versa. One area where the supervised learning classifiers were able to outperform the Wikipedia-based classifier was on the aforementioned Technology articles that the Wikipedia-based classifier kept getting confused on. Since they don't rely on the category hierarchy, they were free from that bias. An example of this is the news article regarding Facebook [SMH1]. This was correctly classified as Technology by all three supervised learning classifiers but was classified under business by the Wikipedia-based classifier. Under the optimal settings, the Wikipedia-based classifier used the following entities:

Entity	Category	Depth
Facebook	Business	4
Mark Zuckerberg	Entertainment	4
Yahoo!	Business	4
Google	Business	4

Table 4: Entities used by the Wikipedia-based classifier

Given the entities, the algorithm classified the article under Business. However for the supervised learning classifiers, the training data allowed them to properly place words like *"facebook", "yahoo!"* and *"google"* nearer to Technology in the case of Centroid and SVM, and greater probability of Technology in the case of NB. Also, other words in the document like *"online"*, *"network"*, *"digital"* and others would have helped in the Technology classification.

On the opposite end, the Wikipedia-based classifier performed well classifying articles for Business, Sports, and Entertainment. The performance in the Sports classification is of particular note because it outperformed the supervised learning classifiers handily. A specific case the other classifiers had trouble with is the news article about an event in the Commonwealth Games [SMH2]. The entities used by the Wikipedia-based classifier under the optimal settings were:

Entity	Category	Depth
Commonwealth Games	Sports	3
Madame Butterfly	Entertainment	4
Susie O'Neill	Sports	4
Australian Institute of Sport	Sports	4
Aussie	Sports	4

Table 5: Entities used by the Wikipedia classifier

Sports easily beat the other possible classifications. One feature of the Wikipedia-based classifier that is shown in these entities is its ability to recognize compound words like "Commonwealth Games" and "Australian Institute of Sport" and use its meaning rather than just using each of the words separately. The words "commonwealth" and "games" have very different meanings to "Commonwealth Games", and the extra semantic information in the compounded word helped the Wikipedia-based classifier classify the article correctly.

Another advantage of the Wikipedia-based classifier is that the news article was Australian-centric. This is in contrast to the Wikipedia articles the supervised learning

classifiers trained on, which can be said to be mostly American or European-centric. Australian-centric articles will constitute a smaller number of the articles in Wikipedia, and a random sampling for generating training data would be getting a smaller number of that still. The Wikipedia-based classifier is able to generalize better to non-American or European test data because it is able to make use of the entire Wikipedia knowledge base for classification, not just a subset which was used for training.

The difference in strengths of each type of classifier can be partially explained by Zipf's law [GZ49]. According to Zipf's law, roughly 20% of the terms account for 80% of the term occurrences in the data, while 20% of term occurrences in the data account for the other 80% of terms in the long tail of the distribution. The first observation supports supervised classification because it means that most of the words in the test data will have been seen previously. At the same time, the second observation implies there will be a lot of infrequent words too, around 80% of the possible words in the test data. Supervised learning classifiers are rarely able to make use of these words, and 20% of the contents of test is a lot to just disregard. The Wikipedia-based classifier is aimed at these words which are at the long tail of Zipf's law distribution.

5. Further Work

During testing it was found that the better the NER system, the better the classification performance. Continued research and improvement to the state of the art NER systems will indirectly improve the results of the classifier as well.

Finally, another possible improvement could be a replacement for Wikipedia as a knowledge base for named entities and category data. Wikipedia provides a huge dataset for harnessing named entities, but it also brings with it its own biases in its category hierarchy. On top of that, the large size of Wikipedia also gives a huge speed penalty in performance. A more compact knowledge base that allows for faster article retrieval and category hierarchy traversal will be a big step forward in the implementation of a more practical classifier that uses named entities.

A better approach in the future would be using named entities as part of an ensemble classifier that makes use of the strengths of named entities with other tools that make up for its weaknesses. Perhaps a combination of a supervised learning algorithm that makes use of named entities only for unseen data during training will be a better classifier in the future.

6. Conclusions

This report has shown that named entities together with a good knowledge base are useful tools in building a better classification algorithm because of the additional semantic information that they provide. The biggest feature of the implemented classification algorithm is using the information in the category hierarchy of Wikipedia to make use of the semantic information available in named entities. This

removes the reliance of a classifier on a huge amount of training data to be able to build an effective model for classification, and in fact pure knowledge based classifiers like the one implemented in this report don't make use of training data anymore at all. Hence, classifiers that rely on knowledge bases are most useful in situations where there are too few training data or none at all for the supervised learning algorithms to be effective.

We have looked at automated text classification and have compared supervised learning methods to a novel knowledge-based method for classification that leverages Wikipedia. Our results show that they tackle the classification problem in entirely different ways and each has its own strengths and weaknesses. Since there are advantages to each approach, future research should investigate how to optimally combine learning based techniques with knowledge-based techniques to leverage the power of learning while reducing the amount of training data required. The end result will hopefully be a more intelligent text classification algorithms that allow computers to automatically organize large amounts of text content to meet user needs.

7. References

[HK00] Eui-Hong Han and George Karypis. Centroid-based document classification: Analysis and experimental results. In *PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 424–431, London, UK, 2000. Springer-Verlag.

[SR02] Fabrizio Sebastiani and Consiglio Nazionale Delle Ricerche. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1– 47, 2002.

[SB87] Gerard Salton and Chris Buckley. Term weighting approaches in auto- matic text retrieval. Technical report, Ithaca, NY, USA, 1987.

[MN98] A. McCallum and K. Nigam. A comparison of event models for naïve bayes text classification, 1998.

[TJ98] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Machine Learning: ECML-98*, pages 137–142. Springer Verlag, 1998.

[JG05] Jim Giles. Internet encyclopedias go head to head. *Nature*, Volume 438, Number 7070, pages 900-901. Nature Publishing Group. 2005

[WK] Wikipedia English website (http://en.wikipedia.org)

[SVM] LIBSVM website (http://www.csie.ntu.edu.tw/~cjlin/libsvm)

[CC01] Chih-Chung Chang and Chih-Jen Lin. LIBSV: a Library for Support Vector Machines. 2001. (http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf)

[CC03] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A Practical Guide to Support Vector Classification. 2003.
(http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf)

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector network. *Machine Learning*, Volume 20, Number 3, pages 273–297. 1995.

[SMH1] Facebook allows cliques, profile backups. *Sydney Morning Herald.* http://www.smh.com.au/technology/technology-news/facebook-allows-cliques-profile-backups-20101007-1689p.html

[SMH2] Coutts lands her aquatic trifecta, with another two golds on horizon. *Sydney Morning Herald.* http://www.smh.com.au/commonwealth-games-2010/comm-games-news/coutts-lands-her-aquatic-trifecta-with-another-two-golds-on-horizon-20101007-169zj.html

[GZ49] George K. Zipf. Human Behavior and Principle of Least Effort. Addison-Wesley. 1949