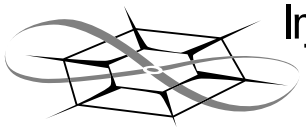
The University of Kansas



Information and Telecommunication Technology Center

Technical Report

Training a Hierarchical Classifier Using Inter-Document Relationships

Susan Gauch, Aravind Chandramouli,
and Shankar Ranganathan

ITTC-FY2007-TR-31020-01

August 2006

Project Sponsor:
National Science Foundation

Abstract

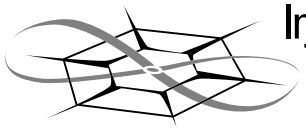
Concept hierarchies, also called taxonomies or directories, are widely used on the World Wide Web to organize and present large collections of Web pages. They were originally developed to help users locate relevant information by browsing. More recently, conceptual search engines such as KeyConcept have been developed that retrieve documents based upon the concepts they discuss in addition to the keywords they contain. Both applications require that documents be classified into appropriate concepts in a conceptual hierarchy. Most classification approaches use flat classifiers that treat each concept as independent, even when the concept space is hierarchically structured. In contrast, hierarchical text classification exploits the structural relationships between the concepts. In this paper, we explore the effectiveness of hierarchical classification for a large concept hierarchy. Since the quality of the classification is dependent on the quality and quantity of the training data, we evaluate the use of documents selected from subconcepts to address the sparseness of training data for the top-level classifiers and the use of document relationships to identify the most representative training documents. By selecting training documents using structural and similarity relationships, we achieve a statistically significant improvement of 39.8% (from 54.5% to 76.2%) in the accuracy of our classifier over that of the flat classifier for a large, 3-level concept hierarchy.

Categories and Subject Descriptors: H.3.1 [**Content Analysis and Indexing**]; I 2.6 [**Learning]**

General Terms: Algorithms

Additional Key Words and Phrases: Hierarchical Text Categorization, Classifier Training

The University of Kansas



Information and Telecommunication Technology Center

Technical Report

Training a Hierarchical Classifier Using Inter-Document Relationships

Susan Gauch, Aravind Chandramouli,
and Shankar Ranganathan

ITTC-FY2007-TR-31020-01

August 2006

Project Sponsor:
National Science Foundation

1 Introduction

1.1 Motivation

As the amount of information on the World Wide Web grows, the task of finding relevant information becomes more difficult. Typical search engines provide many irrelevant results, primarily due to the ambiguity of natural language [Krovetz & Croft 1992] combined with the short length of most Internet searches. For example, the query 'salsa' returns the same results to a person searching for a recipe as to one searching for details about the dance form.

To overcome this problem, the KeyConcept search engine [Ravindran & Gauch 2004] indexes documents by keywords and concepts. This allows the users to restrict their search results to only those documents that match their concepts of interest. In order to be able to retrieve by concept efficiently, documents are indexed by their best matching concepts selected from a pre-existing concept hierarchy. Currently, during indexing, we use a flat classifier to assign newly arriving documents to concepts. This classifier does not take the hierarchical relationships between concepts into account, but rather treats each concept as independent. However, recent work has utilized known hierarchical structure to decompose the problem into a smaller set of problems corresponding to hierarchical splits in the tree [Koller & Sahami 1997]. One first learns to distinguish among concepts at the top level, and then the lower level distinctions are learned only within the appropriate top level of the tree. Earlier studies show the increases in accuracy and efficiency for this approach on small concept hierarchies (2 levels, 150 concepts) [Dumais & Chen 2000], but only recently have researchers been looking at the performance of classifiers on large, hierarchical concept spaces. Yang

[2003] looks at the scalability of flat classification algorithms in terms of efficiency and, in this paper, we demonstrate that hierarchical classification also provided improved accuracy over flat classification for larger, deeper concept hierarchies.

Any classifier's accuracy is affected by the quantity of the documents for each concept used to train the classifiers. We investigate the effect of the amount of training information on the classifier accuracy. However, in addition to the quantity of information, the quality of the training documents is also important. Concept hierarchies tend to have few documents attached at the upper levels. We compare approaches to selecting training documents for the higher-level classifiers by selecting documents from the subconcept training collections. Finally, we evaluate the use of calculating the centroid of the documents in a concept and choosing the documents based on their distance from the centroid to identify the most representative documents for training on each concept.

1.2 Objectives

Our objectives are summarized as follows:

- Develop a top-down, level-based hierarchical classifier and compare it to a flat classifier for a large concept hierarchy.
- Evaluate the criteria for training set selection for hierarchical text classification. In particular, evaluate the number of training documents used per concept, the use of training documents selected from subconcepts, and the effect of centroid distances to select training documents.

1.3 Outline of Paper

The paper is organized as follows: Section 2 discusses work related to text classification including hierarchical techniques. Section 3 details the classifiers used for our experiments. Section 4 discusses our training data, that is, the Open Directory Project collection. Section 5 presents the experiments with the flat classifier. Section 6 discusses our experiments on the hierarchical classifier to validate our approach and analyzes the results. Finally, Section 7 gives the conclusions and points the way to future work.

2 Related Work

2.1 Text Classification

Text classification organizes information by associating a document with the best matching concept(s) from a set of concepts. Classification, requires a predefined set of concepts, also called classes categories, and information describing what types of documents belong in each concept. In general, this knowledge takes the form of a set of documents that have been manually classified into each concept. Classification usually occurs in two phases: the *training* phase in which the classifier learns which features best represent each concept and the *classification* phase during which new, unclassified documents are placed into the best matching concepts. During training, features are extracted from the training documents and these features are used to represent the concept. During classification, features are extracted from the new document and these features are compared to the concept features to identify the best matches.

There has been a tremendous amount of research into classification in general, and text classification in particular. The various approaches differ in how the concepts

and documents are represented, how the features are extracted and weighted, and how the similarity between the documents and concepts is calculated. Although neural networks [Weiner et al.1995; Ng et al. 1997; Ruiz & Srinivasan 1999], rule-based trees [Lu et al. 1999] have all been used as the basis for classification, the vector space model, including Latent Semantic Indexing [Cai & Hofmann 2003], and the probabilistic model have been most widely used, so they will be discussed in more detail.

Probabilistic classifiers use the training documents to calculate probability estimates for each word in the training collection. These estimates represent the probability that, if a new document contains a given word, that document belongs to the particular concept. During classification, words are extracted from the document to be classified and the probability that the document belongs to each concept is calculated. Early work studied pure naïve Bayes classifiers that consider a document as feature vectors of binary, or Bernoulli, variables [Lewis & Ringuette 1994]. These, however, cannot utilize the within-document term frequencies. To improve classification performance, multinomial naïve Bayes classifiers that incorporate this information have been implemented. McCallum and Nigam [1998] compared the performance of the Bernoulli and multinomial Bayes classifiers using text corpora from five different sources. The results indicate that the Bernoulli model performs better on smaller vocabularies while the multinomial model performs better with a larger vocabulary.

There are many vector space approaches to text classification. With the vector space model, the training documents and documents to be classified are represented as multi-dimensional vectors in which each dimension represents a unique term in the training document collection [Salton & McGill, 1983]. The approaches differ in how the

weights for the terms are calculated, how the concept vectors are created, and how the document vectors and concept vectors are compared. One of the most popular approaches is to calculate the term weights using a variant of tf*idf, the term frequency in the document multiplied by the inverse document frequency, a measure of the rarity of the term in the training collection as a whole.

One simple, effective approach is Rocchio classification [Rocchio, 1971] in which the training documents are used to create a single, representative vector for each concept. During classification, the vector for the similarity between the document to be classified and the vectors for each concept is calculated (typically using the cosine similarity metric), and the document is classified into the most similar concept(s). In contrast, with the *k*-Nearest Neighbor (*k*-NN) algorithm [Dasarathy, 1991], a vector is created for each training document and, during classification, the vector for the document to be classified is compared to the vectors for all training documents. The top *k* most similar training documents each provide a single vote for their associated concept, and the document is classified into the concept(s) with the most votes. More recently, Support Vector Machine (SVM) classifiers [Vapnik 2000] have been applied to text classification [Joachims 1998**,** Dumais 1998]. These classifiers begin with the training document vectors used by *k*-NN, but they map these vectors to a higher dimensional space in which the new features are chosen so that they allow the data points in the new space to be linearly separable.

More recently, Guo et al. [2003] developed a new classification approach called kNN-Model that combines *k*-NN and Rocchio. Similar to Rocchio, this approach calculates the generalized vector for each concept (i.e., the centroid of the training

documents). However, similar to *k*-NN, it also represents each concept by the *k* training document for that concept closest to the centroid. This hybrid classifier was compared with a basic *k*-NN classifier, a Rocchio classifier, and an SVM based classifier. They used the same ModApte version of Reuters 21578 for evaluation. Although they did not perform significance testing, they found that their hybrid *k*-NN/Rocchio approach performed slightly better than the Rocchio classifier which, in turn, was slightly better than the *k*-NN classifier. Although their hybrid classifier was outperformed by the SVM classifier, it was considerably more efficient.

Yang and Liu [1999] compared the performance of a variety of classifiers, i.e., SVM, *k*-NN, Linear Least Square Fit [Yang & Pedersen 1997], Neural Networks, and multinomial naïve Bayes. These classifiers were tested on the ModApte version of the Reuters 21578 test collection. After preprocessing, this test collection contained 90 categories, 7,769 training documents, and 3,019 test documents. They found that both SVM and *k*-NN significantly outperformed the other classifiers and that the naïve Bayes classifier significantly underperformed all other classifiers.

As you can see from the above summary, classifiers have performed with varying relative accuracies in different studies. In general, Rocchio, *k*-NN, Naïve Bayes, and SVM have been the most effective and most widely used. In Section 5, we compare these classifiers and select the best as the baseline against which our hierarchical classifier is compared.

2.2 Hierarchical Text Classification

There are basically two approaches used to assign documents to concepts within a hierarchical concept space, namely the big-bang approach and the top-down level based

approach [Sun et al. 2003]. The big bang approach, used by Labrou and Finn [1999], Sasaki and Kita [1998], and Wang et al. [2001], essentially flattens the concept hierarchy and uses a single classifier to assign documents to the best matching category in one step. The predefined concepts are treated in isolation and no use is made of the structure defining the relationships among them. It is essentially flat classification applied to a hierarchical concept space.

McCallum et al. [1998] developed the hierarchical shrinkage model to exploit the hierarchical relationships inside a flat classifier. They make use of a naïve Bayes classifier on three hierarchical collections - the science category of the Yahoo hierarchy, the newsgroup dataset, and the industry sector hierarchy. They make use of a technique called "shrinkage" that smoothes the parameter estimates for a child node using a linear interpolation with all its ancestor nodes. They have shown that this technique improves classification accuracy for a two-level hierarchy containing 80 classes. The biggest improvement occurs when the training data per category is sparse, and the hierarchy has a large number of categories. Baker et al. [1999] also use a similar approach called the hierarchical probabilistic model for topic detection and tracking. They address the problem of sparse data within new classes discovered by topic detection by using data from their siblings in the hierarchy.

In contrast to the hierarchical shrinkage and probabilistic models, Toutanova et al. [2001] use an extended hierarchical mixture model to improve classification for small training sets. They also performed an in-depth comparison of models for automatically generating precise hierarchies for large data sets such as the Web based on minimal training. Another approach is the hierarchical generative model by Gaussier et al. [2002]

for improved classification accuracy by providing a better estimation of word occurrence statistics in leaf nodes using its ancestor nodes in the hierarchy.

In contrast to a classification approach that tries to make a single decision, the top-down level based approach, used by Koller and Sahami [1997], D' Alessio et al. [2000], Dumais and Chen [2000], and Pulijala and Gauch [2004], constructs one or more classifiers at each concept level, and each of these classifiers works as a flat classifier on a subset of the concept space. This hierarchical approach exploits the structure of the concept space during classification.

The basic insight behind hierarchical classification is that concepts that are higher in the hierarchy are farther apart than concepts that are close together further down the hierarchy. Therefore, even when it is difficult to find the precise topic of a document, e.g., color printers, it may be easy to decide whether it is about 'agriculture' or about 'computers'. Building on this intuition, hierarchical classification approaches the problem using a divide and conquer strategy. In the above example, we have one classifier that classifies documents based on whether they belong to 'agriculture' or 'computers'. The task for further classifying within each of these wider concepts is done by separate classifiers within 'agriculture' or 'computers' respectively. The following are some motivations for taking hierarchical structure into account [D'Alessio et al. 2000]

- The flattened classifier loses the intuition that concepts that are close to each other in hierarchy have more in common with each other, in general, than concepts that are spatially far apart. These classifiers are computationally simple, but they lose accuracy because the concepts are treated independently and relationship among the concepts is not exploited.

- Text classification in hierarchical setting provides an effective solution for dealing with very large problems. By treating problem hierarchically, the problem can be decomposed into several problems each involving a smaller number of concepts. Moreover, decomposing a problem can lead to more accurate specialized classifiers.

The test document starts at the root of the tree and is compared to concepts at the first level. The document is assigned to the best matching level-1 concept and is then compared to all subconcepts of that concept. We can use features from both the current level as well as its children to train this classifier. This process continues until the document reaches a leaf or an internal concept below which the document cannot be further classified. One of the obvious problems with top-down approach is that a misclassification at a parent concept may force a document to be mis-routed before it can be classified into child concepts.

3 Approach

Encouraged by promising results with smaller concept hierarchies [Dumais & Chen 2000], we explore the applicability of a hierarchical classification for our large concept hierarchy and compare it to our original flat classifier. Because the quality of classification is dependent on the quantity and quality of the training documents, we evaluate a variety of training strategies for the hierarchical classifier. In particular, we investigate techniques for dealing with the sparseness of training data for the top-level classifiers. These classifiers are particularly important because a wrong decision by the first classifier directs the document to be classified to the incorrect next level classifiers. Our approach is to supplement the training collection for high level concepts with

documents chosen from their subconcept training sets. We look at a variety of approaches for selecting these supplemental documents, specifically looking at the contribution of child and grandchild training documents, selecting documents from a pool with and without regard to the subconcept structure, and using centroid distances to identify the most representative training documents.

In Section 3.1, we briefly describe the different classifiers evaluated to select the flat classifier used as our baseline. Section 3.2 describes how the hierarchical classifier is constructed from the flat classifier. Section 3.3 describes our approach to training document selection for the hierarchical classifier.

3.1 Flat Classifiers

As described in Section 2.1, the Rocchio algorithm [Rocchio, 1971], naïve Bayes [Ferguson, 1973], *k*-NN [Dasarathy, 1991], and Support Vector Machines [Vapnik, 2000] have been shown to perform well for text classification. We compared these high-performing classifiers on our large collection of concepts:

- Rocchio algorithm (local implementation)
- naïve Bayes, *k*-NN (Rainbow [McCallum, 1996])
- Support Vector Machines (LIBSVM [Chang & Lin, 2001])

Since the Rocchio algorithm is a local implementation, we describe it in detail. Our Rocchio formula is identical to that used in the Rainbow package, and produces identical results, but our local implementation creates an inverted index and is thus much faster. The other classifier implementations are described briefly, and interested readers are referred to [McCallum, 1996, Chang & Lin, 2001] for details.

For the Rocchio algorithm, the terms are extracted from the training set and the weight of term *i* in document *j* is calculated as shown in Eq. 1:

$$wt_{ij} = ln(tf_{ic}+1) * idf_i \qquad (1)$$

where

tf_{ic} = the total frequency of term *i* in all training documents for concept *c*

idf_i = the inverse document frequency for term *i*

$$= log \frac{N}{df_i}$$

where

N= the number of training documents

df_i = the number of documents that contain the term *i*

Then, the concept vector is formed by adding the weights of each word in the training documents for that concept. Thus, the weight of each word *i* in a concept *c* is the sum of the weights of word *i* in documents *j*, where *j* is a training document for concept *c*. This equation is shown in Eq. 2.

$$wt_{ic} = \sum_j wt_{ij} \qquad (2)$$

Because not all training documents are the same length, the concepts vary somewhat in the amount of training data. To compensate for this, the term weights in each concept vector are normalized by the vector magnitude, creating unit length vectors. Eq. 3 shows the calculation of nwt_{ic}, the normalized weight of term *i* in concept *c*.

$$nwt_{ic} = \frac{wt_{ic}}{\sqrt{\sum_i wt_{ic} * wt_{ic}}} \qquad (3)$$

During the classification phase, a weighted term vector is generated for the document to be classified in a similar manner. We weight the terms in the document using *ln (tf+1) * idf* and this weight is normalized using the normalization factor described above. The classifier compares this vector to the vectors for each of the concepts using the cosine similarity measure [Salton & McGill 1983]. The results are then sorted to produce a rank-ordered list of matching concepts.

LIBSVM [Chang & Lin, 2001] supports multi-class classification and provides a fast SVM implementation used for text classification [Basu et al. 2003]. Though, it provides support for a variety of kernel functions, we chose to use the linear kernel as it has been shown to work well for text classification [Dumais & Chen, 2000]. The regularization parameter C plays a major role in the classification accuracy of SVM, and this parameter is chosen by performing cross-validation on the training set. The details are described in section 5.1.

The rainbow toolkit [McCallum, 1996] supports classification using *k*-NN and naïve Bayes. The *k*-NN algorithm implemented in rainbow makes use of a distance-weighted *k*-NN, and the performance of the algorithm is governed by the choice of *k*. The value of *k* is chosen by performing cross-validation on the training set, and the details are described in section 5.1. For, the naïve Bayes classifier, we use a multinomial mixture model, and we do not perform feature selection for any of the classifiers.

3.2 The Hierarchical Classifier

Based on encouraging preliminary experiments [Pulijala & Gauch 2004], we built a hierarchical classifier for the concept hierarchy. To do this, we first constructed a set of classifiers, one at each level, using the best classifier obtained as a result of flat classifier

experiments described in Section 5.1. We first classify each test document using the level I classifier and then, based on the top result, reclassify the document using the appropriate level II classifier to find the best level II match. The document is then classified by the top matching level III classifier, and so on, until the bottom of the hierarchy is reached. In our experiments, we built and tested a classifier for a 3-level concept hierarchy that matched documents to the best matching leaf concept. The best-matching higher-level concepts are implicitly identified as the parents and grandparents of the final concept.

3.3 Training Document Selection

The Web has grown to cover such a wide range of topics that concept hierarchies built to organize the content are very large. As we consider the problem of classifying Web documents into large concept hierarchies, we need to carefully select training documents for the classifiers. Since the top-level concepts have few associated training documents, it is difficult to train classifiers for these concepts. We therefore investigate ways of populating their training collections with documents selected from their subconcepts. We look at the impact of the distribution of the selected documents across the subconcept space on the classification accuracy. We evaluate a variety of approaches for selecting the subconcept documents, those that select from a pool of all such documents and those that select the subclass documents paying attention to the subconcept structure.

With any large set of concepts, the boundaries between the concepts are fuzzy. If used for training, documents near the boundaries will add noise and confuse the classifier. We want to eliminate outlier documents, and the words they contain, from the

representative vector for the concept. Thus, we explore the use of calculating the centroids of the candidate training documents for each concept and using the distance of the documents from the centroid in order to identify the most representative training documents for that concept, and evaluate the effect this has on classifier accuracy. The CLUTO Clustering Toolkit [CLUTO 2003] - Release 2.1 is used to calculate the centroid of the candidate training documents. The following clustering parameters were used:

- **Clustering Method**: Partitional Clustering - using bisections.
- **Similarity Function:** Cosine Function
- **Particular clustering criterion function used in finding cluster**: I_2

  where, I_2 is given by :

$$\text{maximize} \sum_{i=1}^{k} \sqrt{\sum_{v,u \in S_i} \text{sim}(v,u)}$$

In the above equation, k is the total number of clusters, S is the total objects to be clustered, S_i is the set of objects assigned to the ith cluster, v and u represent two objects, and sim(v, u) is the similarity between two objects. We assume that all training documents for a given concept belong to a single cluster, and the ***vcluster*** [CLUTO 2003] function and the ***z-scores*** [CLUTO 2003] option are used to calculate the centroid and the distance of the documents from the centroid.

4 Experimental Design

We wish to compare the accuracy of a flat classifier with that produced by a hierarchical classifier in which training documents are selected in a variety of ways.

Section 5 outlines our experimental results using flat classification and Section 6 describes a series of experiments on our hierarchical classifier.

4.1 Test Collection

Because, the Open Directory Project hierarchy [ODP 2004] is readily available for download, it was chosen as the source for classification tree. It is becoming a widely used, informal standard and has been used for hierarchical classification experiments [Dhillon et al. 2002, Dekel et al. 2004]. As of December 2004, the Open Directory had more than 590,000 concepts created by over 66,000 editors. With such a fine granularity, subtle differences between certain concepts may be apparent to a human but indistinguishable to a classification algorithm. In order to capture broader differences between documents, documents are classified into concepts from the top three levels only, although training data from the top four levels was used in some experiments. A part of the ODP hierarchy is shown in Figure 1.

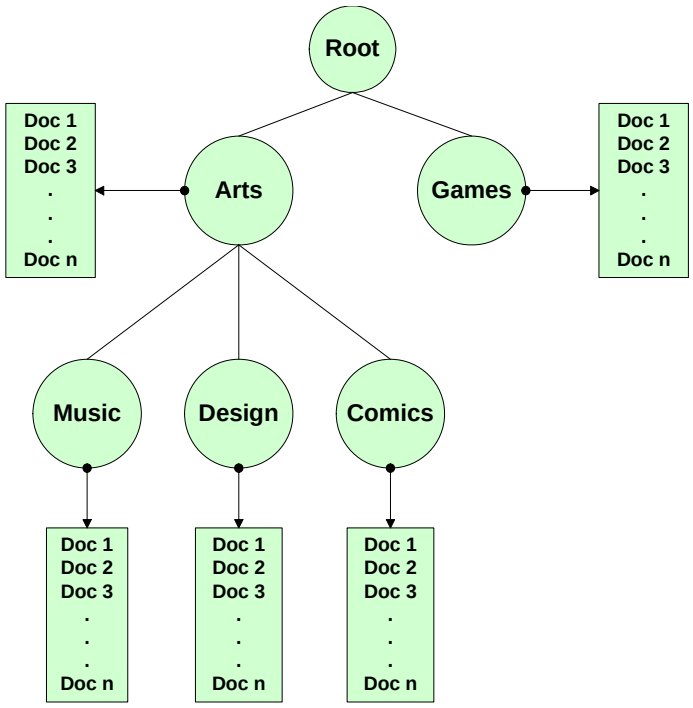


Figure 1. Part of the ODP Hierarchy

Experiments with training set sizes reported in [Gauch et al. 2004] showed that the classifier performed at its peak when trained using 30 documents per concept. Because we wished to evaluate our algorithm on a truly large hierarchy, we made a local copy of the ODP collection that contained all of the first 4 levels of the hierarchy and downloaded a maximum of 100 associated documents for each concept. We then pruned out any level III concepts, and their child subconcepts, that had fewer than 31 training documents (30 for training, 1 for testing). This created a subset of the ODP that contained all 15 level I concepts, 358 level II concepts, 1,211 level III concepts and 10,132 level IV concepts.

For testing purposes, we randomly selected 1 document from each of 1,000 different level III concepts that were withheld from training. Since we know the concept from which each document was selected, we can evaluate the accuracy of our classifier against "truth" by measuring how often the classifier assigns the test documents to the concepts from which they originally came.

5 Flat Classification Experiments

This section describes our experiments to with flat classifiers. Experiment 0 establishes the accuracy of the baseline against which the hierarchical classifier will be compared. We first compare Rocchio, *k*-Nearest Neighbors, Support Vector Machines (SVM), and naïve Bayes on our dataset. The best performing classifier is then used for the experiments with the hierarchical classifier. Experiment 1 shows the effect of using the centroid distances to select the training documents for the flat classifier chosen from experiment 0.

5.1 Experiment 0: Determining the Flat Classifier Baseline

We first establish a baseline level of performance with the flat classifier built using the Rocchio classifier, *k*-NN, SVM and naïve Bayes as described in section 3.1. Since automatic classification algorithms are often asked to place documents in a single concept, all evaluations were made comparing the accuracy of the top-ranked result only.

We performed a five-fold cross-validation on the training set to determine the best choice of parameters for *k*-NN and linear SVM. The values $k \in \{1, 10, 20, 30, 40, 50, 60\}$ for the *k*-NN classifier and $C \in \{0.01, 0.1, 1, 10, 100\}$ for the SVM classifier were tried. The best performing parameter is used for these algorithms and the results obtained are shown in Table 1.

	Rocchio	*k*-NN k=40	SVM C= 0.01	Naïve Bayes
Accuracy	54.45%	24.24%	0.18%	27.27%

Table 1: Accuracy of the different flat classifiers on the test collection

The results in Table 1 show that the Rocchio classifier performed the best and the SVM classifier performed the worst, while naïve Bayes was slightly more accurate than *k*-NN. Since most studies find that SVM outperforms other classifiers, these results are somewhat surprising. We believe that the poor performance of SVM is due to the high dimensionality of the data set. Because there are so many concepts, and so many training documents, our vectors contain an average of 10,859 features per concept. Based on these results, we use the Rocchio classifier as our baseline for the rest of the experiments.

5.2 Experiment 1: Using Centroid Distances to Select the Training Set for Flat Classification

The goal of this experiment is to see if centroid distances can be used to select a better set of training documents and thereby improve the accuracy of the flat classifier. Rather than randomly selecting documents from each concept's associated documents,

we calculated the centroid of the document collection and used the distance of the centroid from the documents to identify the documents that might best represent the concepts. First, for each concept, we calculated the centroid for the set of all associated non-test documents. We then evaluated a variety of approaches by which to select the training documents for each concept. The first approach concentrates on using the documents that have the most in common and the other two approaches use the documents that provide the best breadth of coverage.

- **Method A:** We choose 30 documents that are closest to the centroid.
- **Method B:** We choose 30 documents that are farthest from the centroid.
- **Method C:** We choose 30 documents that are farthest from each other.

Selection Algorithm	Accuracy	Improvement Over Baseline
Baseline (Random)	54.5%	---
Close to Centroid	55.7%	1.2%
Far from Centroid	54.9%	0.5%
Far from Each Other	44.4%	(10.1)%

Table 2: Accuracy of the flat classifier trained on documents selected using centroid distances

The results presented in Table 2 show that selecting documents closest to the centroid from each concept yields the highest accuracy, 55.7%, a 1.2% improvement over the baseline. This provides only a modest increase in accuracy, leading us to explore the use of hierarchical classification for larger potential gains.

6 Hierarchical Classification Experiments

This section describes our experiments with the hierarchical classifier. Section 6.1 describes a series of experiments that pools the associated documents for a concept with those from child and grandchild concepts. The training documents for each concept are then selected from this pool of documents with and without considering centroid distances. The distribution of the training documents across the concepts and subconcepts is not taken into consideration. Section 6.2 describes a different training document selection algorithm that selects documents uniformly across the subconcepts of the concept being trained. Based on the results obtained from the above experiments, we created a generic training algorithm for classifiers. This is outlined in section 6.3 along with the validation performed of this algorithm on a new set of test documents collected by randomly choosing one document that has not been used for either training or testing in the previous experiments from each of the level III concept. For each of the experiments described, we built a level I classifier, 15 level II classifiers, and 358 classifiers for level III that were used to assign documents to one of 1,211 level III concepts.

6.1 Using Pooled Documents For Training

In this section, we describe a set of experiments that select training documents from collections of candidate documents that are pooled together. The candidate document pools always contain each concept's associated documents. Because the number of documents associated with the for the level I and level II concepts is very low, we then evaluate the effects of adding the documents associated with subconcepts to the pool of candidate documents. We compare selecting the training documents from the

pool randomly with calculating the centroid of the documents in the pool and selecting those closest to the centroid. The effect of each algorithm on the classification accuracy for levels I, II and III is evaluated by Experiments 2, 3, and 4 respectively.

6.1.1 Experiment 2 Level I Classification Accuracy

This experiment evaluates the level I classification accuracy when the classifier is trained using pooled collections of level I documents only, levels I and II pooled together and levels I, II, and III pooled together. Each pool is created by combining all associated documents. We then select 10 through 90 documents for training, either randomly or selecting the documents closest to the centroid.

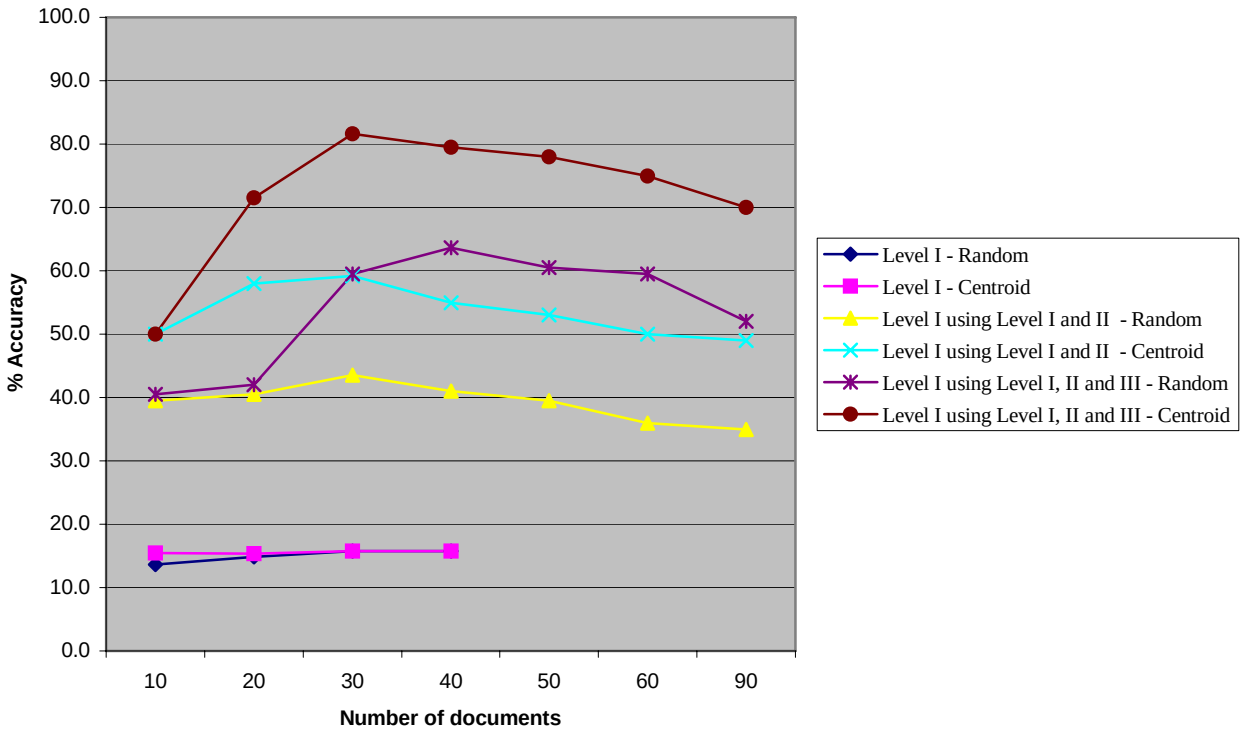


Figure 2. Level I decision accuracy

Figure 2 shows the classification accuracy for the level I classifier trained in a variety of ways. For all methods, the performance peaks with 30 or 40 training documents. When trained on the associated level 1 documents alone because of the sparseness of the training data, the classifier performs very poorly (13.6% random selection, 15.8% selecting near the centroid). Training improves as the candidate pool increases, performing best when documents from levels I, II, and III are pooled. With this pool, the highest accuracy for randomly-selected documents is 63.6% when trained using 40 documents. This is a 307% improvement (48.8% absolute) over documents selected randomly from the level I pool alone.

When we select the 30 documents closest to the centroid, we see a further improvement to 81.6% accuracy, a 396% improvement (63% absolute) when compared to selecting the 30 documents closest to the centroid from the level I documents alone. From these results, we conclude that we get the most accurate level I decision when we train the classifiers on documents pooled from levels I, II, and III from which we select the 30 documents closest to the centroid.

One surprising observation is that, as the number of training documents increases beyond 30 or 40 per concept, the accuracy decreases. We attribute this to the fact that choosing documents close to the centroid selects the best representative documents and that, as documents are added, more peripheral documents are included. Even when centroid distances are not used, adding extra documents increases the size of the vocabulary (i.e., features) used to represent the concept and the resulting increase in noise decreases the accuracy and increases the vocabulary overlaps between concepts.

6.1.2 Experiment 3 Level II Classification Accuracy

Experiment 3 essentially reproduces Experiment 2 for level II classification. Thus, it investigates the effect of centroid distances on a set of candidate training documents created by pooling documents at levels II and III.

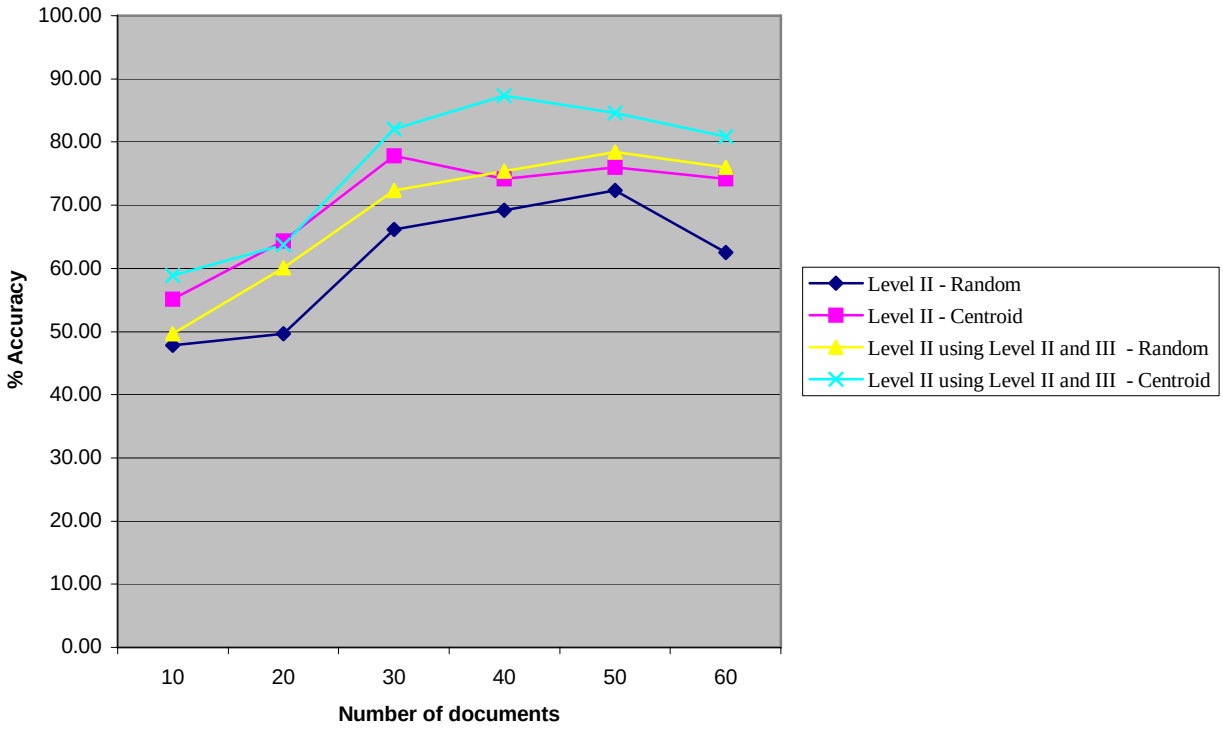


Figure 3. Level II Decision Accuracy

With our best level I classifier, 816 (81.6% of the 1000) test documents were sent to the correct level II classifier. Figure 3 shows the accuracy of the level II classification for these 816 test documents. From this figure, we see that the classifier is more accurate when trained on pooled level II and III documents rather than on level II documents alone. 78.1% versus 87.3%. Thus, including level III documents in the training of level II concepts improves the accuracy of the level II classifiers, even though most level II concepts have over 30 training documents of their own.

We also observe that the accuracy for training documents selected near the centroid is higher then documents selected at random. In particular, the maximum

accuracy for a randomly selected training set is 78.4% observed when 50 documents are used for training. However, when documents closest to the centroid are used to train the classifier, the best accuracy of 87.3% with 40 training documents. Thus, there is an improvement of 11.3% (8.9% absolute) when centroid distance is used to choose the training documents. Given that 184 documents were misclassified by the level I classifier, this produces an overall cumulative accuracy of 71.3% after 2 levels, i.e., 713 of the 1,000 test documents are assigned to the correct level II concept.

6.1.3 Experiment 4 Level III Classification Accuracy

At the end of the experiments for level II, out of 1000 initial test documents 713 have correctly identified their level II concept. Since all test documents are drawn from level III, the last step is to measure how many of these documents ultimately make it to their true concepts. All level III concepts contain at least 31 associated documents. Thus, we expect to be able to train them successfully using only their own documents, without augmenting the training collection with documents associated with child concepts. To validate this, we train the level III classifiers using 10 through 60 training documents selected randomly and by their closeness to the centroid.

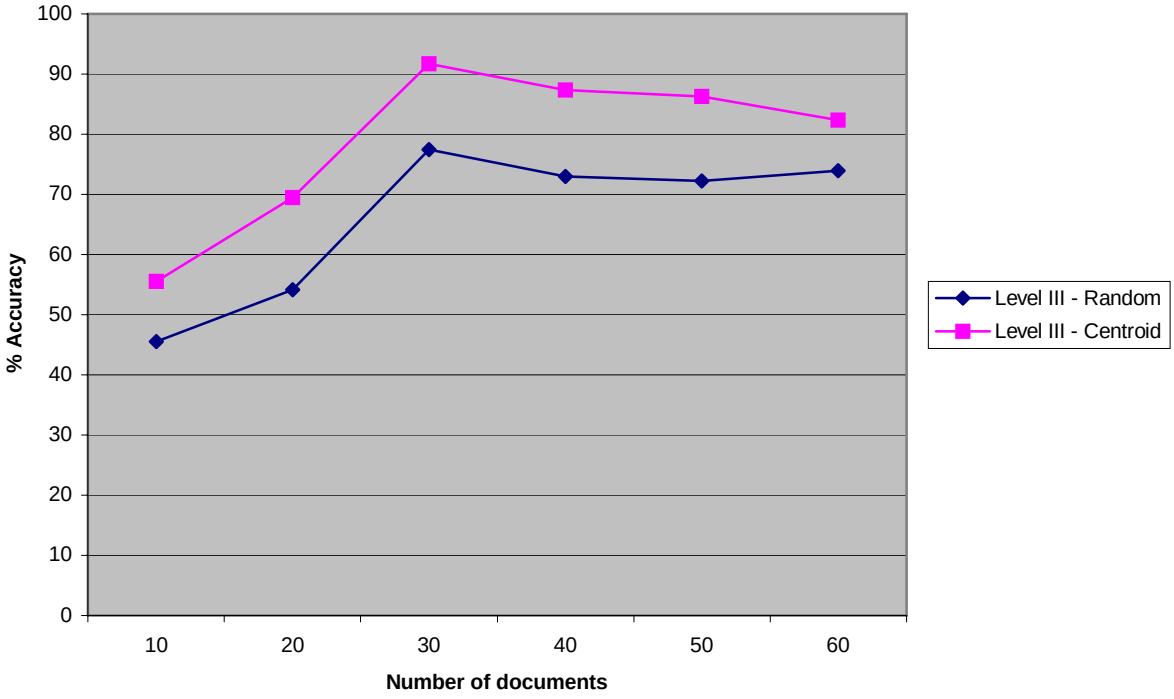


Figure 4. Level III decision accuracy using only documents from level III.

Of the 713 documents that made it to the correct level III classifier, the randomly trained level III classifier assigns 552 to their correct concept. This means that the level III classifier is 77.4% accurate (55.2% cumulative accuracy). In contrast, when the level III classifier is trained with the 30 documents closest to the centroid, 654 documents are assigned to the correct concept. Thus, the level III classifiers are 91.7% accurate (65.4% cumulative accuracy) when centroid distance is used to identify the training documents, a relative improvement of 18.5% (14.3% absolute) over the randomly trained classifiers.

6.2 Using Distributed Documents for Training

The experiments conducted in section 6.1 selected training documents from a set of candidates formed by pooling documents associated with a given concept and its subconcepts. The documents selected were chosen based on their distance from the centroid of the pooled training documents. However, this algorithm did not take into

account the distribution of the selected documents across the subconcept space. This set of experiments evaluates the use of the hierarchical structure during the selection of the training documents. By selecting a specific number of documents per concept or subconcept, the training set should be representative of the breadth of the concept. Based on the results in the previous experiments, we select the subconcept representatives as those nearest the centroid in all experiments reported here.

6.2.1 Experiment 5 Level I Classification Accuracy

For the crucial level I decision, we conduct three different experiments using documents from just level I, levels I and II, and levels I, II and III to train the classifier. We vary the number of documents selected per concept from 1 through 4. In each of the following experiments, the number of documents used for training varies. Because the number of documents per concept is varied between 1 and 4 documents per concept, as more levels are used for training, more subconcepts are added, and thus more training documents.

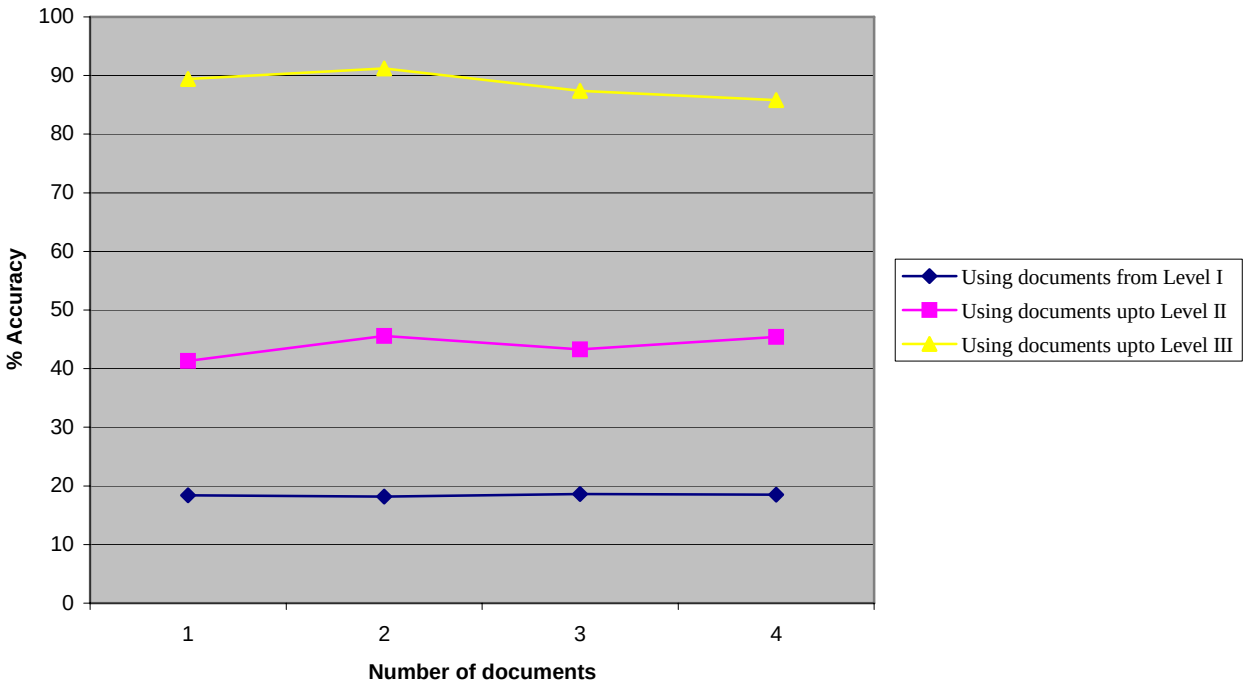


Figure 5. Level I decision using documents closest to the centroid from each concept

From Figure 5, we see that we achieve the best accuracy, 91.2%, when we select the 2 training documents nearest the centroid for each concept and its subconcepts down to level III. This compares favorably with earlier work [Pulijala & Gauch 2004] on the same collection that achieved a maximum accuracy of 79% at level I when selecting documents at random for each concept/subconcept.

Interestingly, when only level I documents are used, this is the same approach as reported in Figure 2, with far fewer documents selected for training. However, the accuracy is almost identical, just under 20%, when only 1 document is used for training as compared to up to 40 documents in Experiment 2. Since the experimental results in Experiments 2 through 5 show a drop off in accuracy as more documents are added, we attribute the improved performance of the classifier to the inclusion of subconcept and

sub-subconcept representative documents rather than due to the increase in number of documents total. In fact, as the number of total training documents used increases by including more representative documents per concept from 1 to 4, we see little change in the accuracy of the classifier. There appears to be a slight peak with 2 documents per concept, then a decrease as more documents are added. We attribute this decrease to the inclusion of noise and increase in overlap between concepts.

6.2.2 Experiment 6 Level II Classification Accuracy

Experiment 6 essentially reproduces Experiment 5 for level II classification. Thus, it investigates the effect of selecting the set of training documents evenly across the subconcept space, using centroid distances to identify the training documents nearest the centroid for each concept. We report the accuracy on the 912 documents that were classified correctly at level I using the most accurate training method, i.e., 2 documents per concept closest to the centroid and all concepts down to level III. Figure 5 shows the Level II classification accuracy obtained by training the classifiers on documents from level II concepts alone versus using documents from levels II augmented with the appropriate level III subconcepts.

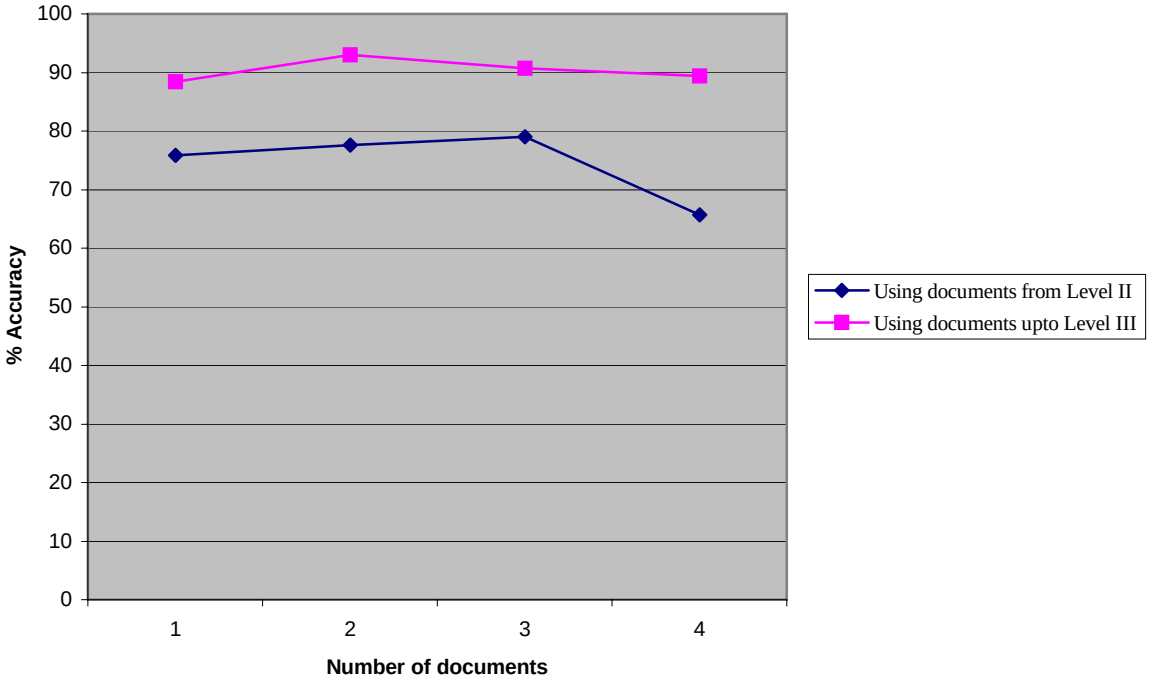


Figure 6. Level II decision using documents closest to the centroid from each concept

It is clear from Figure 6 that we observe higher accuracy when we use documents from levels II and III compared with that achieved training the level II concepts on documents from level II alone. We achieve the highest accuracy of 92.9% when using 2 documents per concept. Given that 88 documents were misclassified by the level I classifier, this produces an overall cumulative accuracy of 84.8% after 2 levels, i.e., 848 of the 1,000 test documents are assigned to the correct level II concept. This compares favorably with a cumulative accuracy of 71.3% when the representative documents are chosen at random [Pulijala & Gauch 2004].

6.2.3 Experiment 7 Level III Classification Accuracy

In this experiment we train the level III classifiers using only data from level III and then by using data from level III as well as level IV. We include level IV documents in this experiment to see if the inclusion of training documents from subconcepts can

improve the level III classifiers. In both the above cases, we select the documents that are closest to the centroid and vary the number of documents to find the combination that gives us the best observed accuracy.

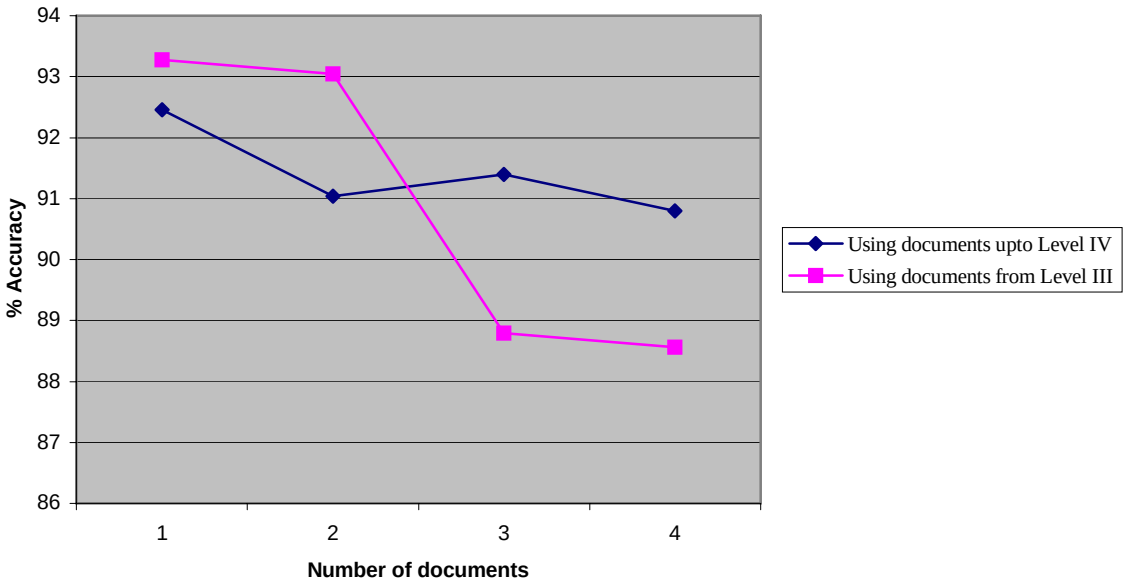


Figure 7 Level III decision using documents closest to centroid from each concept

From Figure 7, we can see that there is not much difference in the accuracy of the level III classifiers when documents from the subconcepts are included. In fact, we observe higher accuracy (93.2%) when the classifiers are trained on level III documents alone compared to when level IV documents are also used (92.4%). Given that 152 documents were misclassified by the level I and II classifiers, this produces an overall cumulative accuracy of 79.1% after 3 levels, i.e., 791 of the 1,000 test documents are assigned to the correct level III concept. This compares favorably with a cumulative accuracy of 70.1% when the representative documents are chosen at random [Pulijala & Gauch 2004].

6.3 Validation

Based on the results from sections 6.1 and 6.2, we devised a simple training algorithm for our hierarchical classifier. We train the classifier for each concept by selecting two training documents from each concept and subconcept down to level III. To validate this straightforward training algorithm, we classify 400 new documents that the classifier has not seen before. We again compare the results of using the documents closest to the centroid to train the classifier versus training the classifier using two randomly selected documents per concept. The results are given in Figure 7.

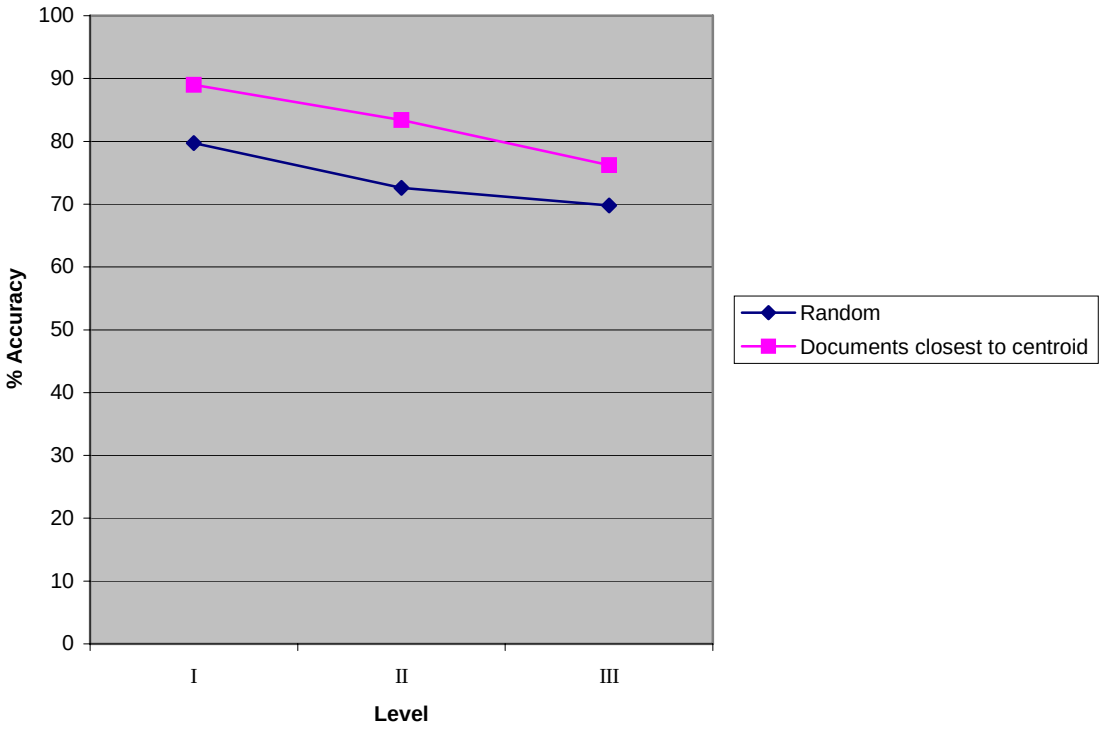


Figure 8. Cumulative Classification Accuracy on Validation Documents

In Figure 8, we observe that selecting the documents closest to the centroid improves the level I classifier's accuracy from 79.7% to 89% and the level II accuracy from 72.6% to 83.4%. The cumulative level III accuracy is 76.2% when the documents closest to the centroid are selected versus 69.8% for random selection. We performed a

two-tailed t-test with alpha value=0.05. We achieve a statistically significant improvement (*p = 3.23E-05)* of 9.1% (6.4% absolute) in our hierarchical classifier.

6.4 Discussion

Table 3 summarizes the results from our experiments. The flat classifier trained on randomly selected documents produces an accuracy of 54.5%, which is our baseline. Using centroid distances to select training documents for the flat classifier produces a minor improvement to 55.7%. Hierarchical classification provides nearly the same accuracy at 55.2% when training documents are pooled from all three levels. This is further increased when we select the 30 documents closest to the centroid (40 for level II classifiers) for training. By requiring that the selected training documents be distributed evenly across the subconcept space, the accuracy further improves to 70.1% when they are selected randomly for each concept and 79.1% when the documents selected are closest to the centroid.

We validated our selection criteria using a new set of testing documents, confirming that we can achieve high accuracy, 76.2% on a large concept hierarchy using hierarchical classification. When the documents selected for each subconcept are closest to the centroid, we see a statistically significant improvement compared to when they are selected randomly.

Selection Algorithm	Accuracy
Flat classifier, pooled, (random)	54.5%
Flat classifier, pooled, (centroid)	55.7%
Hierarchical classifier, pooled (random)	55.2%

Hierarchical classifier, pooled (centroid)	65.4%
Hierarchical classifier, distributed (random)	70.1%
Hierarchical classifier, distributed (centroid)	79.1%
Hierarchical classifier, distributed (random) Validation	69.8%
Hierarchical classifier, distributed (centroid) Validation	76.2%

Table 3. Results Summary

7 Conclusions

In this paper, we compare a flat classification approach to hierarchical classification for a large, three-level concept hierarchy. We present a study of training algorithms for hierarchical classification that evaluates the use of subconcept documents for training high-level concepts. In particular, we evaluate the use of pooling the documents and selecting documents from the pool at random versus selecting the documents nearest the centroid from the pool. We also investigate a training document selection algorithm that selects the training documents evenly across the subconcept space.

We conclude that, hierarchical classification is much more accurate than flat classification, 79.1% versus 55.7% in the best case for each. We also found that calculating the centroid of the candidate documents and selecting those near the centroid improved classification accuracy in all experiments. The relative improvements ranged from a minimum of 2.2% when training the flat classifier using pooled documents to a maximum of 18.5% when training the hierarchical classifier using pooled documents. We found that, when training the top-level hierarchical classifiers, including documents

from subconcepts provides a large improvement in the accuracy. The most dramatic improvement occurs with the level I classifier when it is trained on pooled documents nearest the centroid. When it is trained on only its associated documents, the classifier achieves a very poor 15.9% accuracy. However, when selecting from pooled documents from levels I through III nearest the centroid, the accuracy jumps to 81.6%. Finally, we found that selecting the training documents so that they are distributed evenly across the subconcept spaces, rather then from a pool of all candidate documents, further improves the level I accuracy improved to 91.2%.

The ultimate evaluation, though, is how often the level III test documents are assigned to the correct concept. The best case for flat classification was 55.7%, whereas using pooling and distribution to select the training documents, the hierarchical classifier achieved an accuracy of 79.1%. This is a tremendous improvement of 42% (23.4% absolute). These results were validated with a new set of test documents and a similar accuracy of 76.2% was obtained.

In the future, we would like to explore the use of Support Vector Machines as the classifier for the hierarchical classifier. Although it performed poorly on a large dataset with a large number of concepts, the accuracy might improve when used level by level, since each classifier needs to decide between a much smaller number of concepts. The smaller number of concepts, combined with feature selection, may allow SVM to further improve our hierarchical classifier.

One drawback to our current hierarchical classifier is that it has no way to recover from an incorrect level I decision. The flat classifier can naturally assign documents to multiple categories and, although many applications require the assignment of a

document to a single category, we are working on improving our hierarchical classifier to allow it to do soft categorization. We are developing a more sophisticated algorithm that simultaneously explores more than one branch in the conceptual hierarchy. In addition to potentially improving accuracy, this could also allow documents to be classified into more than one concept.

References

BAKER, L.D., HOFMANN, T., MCCALLUM, A., AND YANG, Y. A hierarchical probabilistic model for novelty detection in text. http://www.cs.umass.edu/~mccallum/papers/tdt-nips99s.ps

BASU, A., WATTERS, C. R., AND SHEPHERD, M. A. 2003. Support Vector Machines for Text Categorization. In *Proceedings of the 36th Hawaii International Conference on System Sciences* (January), 103.

CAI, L. AND HOFMANN, T. 2003. Text Categorization by Boosting Automatically Extracted Concepts. In *Proceedings of the 26th ACM-SIGIR International Conference on Research and Development in Information Retrieval* (July/August), 182-189.

CHANG, C. AND LIN, C. 2001. LIBSVM. A library for support vector machines. Website: http://www.csie.ntu.edu.tw/~cjlin/libsvm

CLUTO. 2003. Cluto. Website: http://www-users.cs.umn.edu/~karypis/cluto/index.html

DASARATHY, B. V. 1991. Nearest Neighbor (NN) norms: NN Pattern Classification Techniques. *McGraw-Hill Computer Science Series. IEEE Computer Society Press*, Las Alamitos, California.

DEKEL, O., Keshet, J., AND SINGER, Y. 2004. Large margin hierarchical classification. *Proceedings of the twenty-first international conference on Machine learning* (July), 27.

DHILLON, I. S., MALLELA, S., AND KUMAR, R. 2002. Enhanced word clustering for hierarchical text classification, *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (March), 191-200.

DUMAIS, S. 1998. Using SVMs for text categorization. *In IEEE Intelligent Systems Magazine, Trends and Controversies, Marti Hearst, ed., 13(4)* (July/August).

DUMAIS, S., AND CHEN, H. 2000. Hierarchical classification of Web content. In *Proc. Of the 23rd ACM International Conference on Research and Development in Information Retrieval*, 256-263.

D'ALESSIO, S., MURRAY, K. SCHIAFFINO, R., AND KERSHENBAUM, A. 2000. The effect of using hierarchical classifiers in text categorization. In *Proceedings Of the 6th International Conference "Recherched'Information Assistee par Ordinateur"*, 302-313.

FERGUSON, T.S. 1973. A bayesian analysis of some nonparametric problems. *Annals of Statistics*, 209-230.

GAUCH, S., MADRID, J. M., INDURI, S., RAVINDRAN, D., AND CHADALAVADA, S. 2004. KeyConcept : A Conceptual Search Engine. *Information and Telecommunication Technology Center, Technical Report : ITTC-FY2004-TR-8646-37,* University of Kansas.

GAUSSIER, E.. GOUTTE, C., POPAT, K., AND CHEN, F. 2002. A Hierarchical Model for Clustering and Categorising Documents. In *Proceedings of the European Colloquium on IR Research* (March), 299.

GUO, G., WANG, H., BELL, D., BI, Y., AND GREER, Y. 2003. Using kNN Model-based Approach for Automatic Text Categorization, In *Proc. of ODBASE'03, the 2nd International Conference on Ontologies, Database and Applications of Semantics, LNCS 2888*, 986-996.

JOACHIMS, T. 1998. Text Categorization with Support Vector Machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning(ECML)*, 137-142.

KOLLER, D., AND SAHAMI, M. 1997. Hierarchically classifying documents using very few words. In *Proceedings of the 14th International Conference on Machine Learning*, (July), 170-178.

KROVETZ, R., AND CROFT, B. W. 1992. Lexical Ambiguity and Information Retrieval. *ACM Transactions on Information Systems*, 10(2), (April), 115-141.

LABROU, Y., AND FININ, T. 1999. Yahoo! As An Ontology – Using Yahoo! Concepts To Describe Documents. In *Proceedings of the 8th International Conference On Information Knowledge Management (CIKM)* (November), 180-187.

LEWIS, D. D., AND RINGUETTE, M. 1994. Comparison of two learning algorithms for text categorization. In *Proceedings of the Third Annual Symposium on Document analysis and Information Retrieval (SDAIR' 94)* , 81-93.

LU, F., JOHNSTEN, T., RAGHAVAN, V., AND TRAYLOR, D. 1999. Enhancing Internet Search Engines to Achieve Concept-based Retrieval. *In Proceeding of Inforum'99*.

MCCALLUM, A. K. 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. Website: http://www.cs.cmu.edu/~mccallum/bow

MCCALLUM, A., AND NIGAM, K. 1998 A Comparison of Event Models for Naive Bayes Text Classification, In *AAAI-98 Workshop on Learning for Text Categorization*, 41-48.

MCCALLUM, A., ROSENFELD, R., MITCHELL, T., AND NG, A. 1998. Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the Fifteenth International Conference on Machine Learning*, 359-367.

NG, H. T., GOH, W. B., AND LOW, K.L. 1997. Feature selection, perceptron learning, and a usability case study for text categorization. In *20th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR' 97)* (July), 67-73.

ODP. 2004. Open Directory Project. Website: http://dmoz.org

PULIJALA, A., AND GAUCH, S. 2004. Hierarchical Text Classification. *International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA* (July).

RAVINDRAN, D., AND GAUCH**,** S. 2004. Exploiting Hierarchical Relationships in Conceptual Search**,** *Proceedings of the 9th International Conference On Information Knowledge Management (CIKM),* (November), 238-239.

ROCCHIO, J. 1971. Relevant feedback in information retrieval. In G. Salton (ed.). *The smart retrieval system - experiments in automatic document processing, Englewood Cliffs, NJ.*

RUIZ, M., AND SRINIVASAN, P. 1999. Hierarchical Neural Networks For Text Categorization. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (August), 281-282.

SALTON, G., AND MCGILL, M. J. 1983. Introduction to Modern Information Retrieval. *McGraw-Hill*, New York, NY.

SASAKI, M., AND KITA, K. 1998. Rule-based text categorization using hierarchical concepts. In *Proceedings of the IEEE Conference on Systems, Man, and Cybernetics*, 2827-2830

SUN, A., LIM, E., AND NG, W. 2003. Performance Measurement Framework for Hierarchical Text Classification. *Journal of the American Society for Information Science and Technology, 54(11),* 1014-1028.

TOUTANOVA, K., CHEN, F., POPAT, K., AND HOFMANN, T. 2001. Text Classification in a Hierarchical Mixture Model for Small Training Sets. *In Proceedings of the 10th International Conference on Information and Knowledge Management* (November),105-113.

VAPNIK, V. 2000. The Nature of Statistical Learning Theory, *2nd Edition, Springer*, New York.

WANG, K., ZHOU, S., AND HE, Y. 2001. Hierarchical classification of real life documents. In *Proceedings of the 1st SIAM International Conference on Data Mining.*

WIENER, E.D., PEDERSEN, J. O., AND WEIGEND, A.S. 1995. A neural network approach to topic spotting. In *Proceedings of {SDAIR}-95, 4th Annual Symposium on Document Analysis and Information Retrieval*, 317-332.

YANG, Y., AND PEDERSEN, J. P. 1997. Feature selection in statistical learning of text categorization. *In The Fourteenth International Conference on Machine Learning* (July), 412-420.

YANG, Y. AND LIU, X. 1999. A re-examination of text categorization methods, In *Proceedings of the SIGIR 1999* (August), 42-49.

YANG, Y. 2003. A scalability analysis of classifiers in text categorization. In *Proceedings of SIGIR-03, 26th ACM International Conference on Research and Development in Information Retrieval* (July), 96-103.