Hierarchical Text Categorization Using Neural Networks

Miguel E.Ruiz and Padmini Srinivasan
School of Library and Information Science,
The University of Iowa,
3087 Main Library, Iowa City IA 52242-1420

Abstract.

This paper presents the design and evaluation of a text categorization method based on the Hierarchical Mixture of Experts model. This model uses a divide and conquer principle to define smaller categorization problems based on a predefined hierarchical structure. The final classifier is a hierarchical array of neural networks. The method is evaluated using the UMLS Metathesaurus as the underlying hierarchical structure, and the OHSUMED test set of MEDLINE records. Comparisons with an optimized version of the traditional Rocchio's algorithm adapted for text categorization, as well as flat neural network classifiers are provided. The results show that the use of the hierarchical structure improves text categorization performance with respect to an equivalent flat model. The optimized Rocchio algorithm achieves a performance comparable with that of the hierarchical neural networks.

Keywords: Automatic Text Categorization; Applied Neural Networks; Hierarchical Classifiers

1. Introduction

A system that performs text categorization aims to assign appropriate labels (or categories) from a predefined classification scheme to incoming documents. These assignments might be used for varied purposes such as filtering, or retrieval. Given the rapid growth of information, automatic text categorization is an important goal. This task has been explored by many researchers in the Information Retrieval (IR), and the Artificial Intelligence (AI) communities. Different approaches such as decision trees (ID3) [25], rule learning [1], neural networks [27, 39], linear classifiers [19], K-nearest neighbor (KNN) algorithms [43], support vector machine (SVM) [11], and Naive Bayes methods [17, 21] have been explored. Interestingly it is only recently that researchers [14, 22, 24, 27, 38] have tried to take advantage of the hierarchical structure available in certain classification schemes, e.g. Medical Subject Headings (MeSH), Yahoo! topic hierarchy.

The hierarchical structure of a classification scheme reflects relations between concepts in the domain covered by the classification. The hierarchy typically encodes a set inclusion relation, also called IS-A relation, between category members. For example, in a classification of living things the set of animals includes the set of fish which includes the set of

trouts. Thus a directional (IS-A) hierarchical link connects the narrower concept of 'trout' to 'fish' which in turn has a similar connection to 'animal'. The IS-A relation is asymmetric (e.g. all dogs are animals, but not all animals are dogs) and transitive (e.g., all pines are evergreens, and all evergreens are trees; therefore all pines are trees). We believe that given such a hierarchical classification the properties used to categorize an entity as a 'trout' must be more closely related to the properties for the class 'fish' in comparison to the properties corresponding to the class 'evergreens'. We suggest that by ignoring the conceptual configurations that accompany classification schemes one may limit the potential of text categorization methods. Thus our primary research goal is to explore the hypothesis that a text categorization procedure capable of exploiting the conceptual connections between categories is more effective than a procedure that is not designed to exploit such information.

More specifically, we explore a categorization strategy designed to exploit the hierarchical structure underlying the UMLS (Unified Medical Language System) Metathesaurus [26] and test its effectiveness using the OHSUMED test collection [9]. Our hierarchical classifier is inspired by the Hierarchical Mixture of Experts model proposed by Jordan and Jacobs [13]. Built as a collection of neural networks, our classifier should scale to larger collections of categories and documents because it divides the categorization problem into a set of related sub-problems.

We present experiments that explore the value of our hierarchical classifier in comparison to a non hierarchical (flat) baseline classifier and a state-of-the-art implementation of the Rocchio classifier. Part of our research goal is to study feature selection as well as the selection of training examples within the overall goal of exploring hierarchical classifiers.

In section 2 we present the theoretical background of the Hierarchical Mixture of Experts model and in section 3 the details of its implementation. Sections 4 and 5 explain the different methods used for feature selection and training set selection while sections 6 and 7 describe the experimental collection and the evaluation measures respectively. Section 8 presents the details of the experiments performed. Section 9 presents an analysis of results while section 10 compares our approach with other work in hierarchical text categorization. Section 11 presents our conclusions and future research plans.

2. Theoretical Framework

The *Hierarchical Mixture of Experts* (HME) model is a supervised feedforward network that may be used for classification or regression [13]. It is based on the principle of "divide and conquer" in which a large problem is divided into many smaller, easier to solve problems whose solutions can be combined to yield a solution to the complex problem. Various methods for subdividing large problems have been proposed. The simplest approach is to divide the problem into sub-problems that have no common elements, also called a "hard split" of the data. The optimum solution of the smaller problems can then be chosen on a "winner-takes-all" basis. Classification and Regression Trees (CART) [3] are based on this principle. Stacked Generalization [40] also uses a hard split of the data and a weighted sum with weights derived from the performance of the smaller problems in their partition space. In contrast, HME divides the large problem into sub-problems that can have common elements – a "soft split" of the elements into a series of overlapping clusters. The outputs of the simple problems are combined stochastically to obtain a global solution. The model has two basic components: gating networks and expert networks. The gating networks, located at the intermediate level nodes of the tree, receive the input $\mathbf{x^{(m)}}$ (a vector $\mathbf{x}$ of m input features representing a document) and produce scalar output that weights the contribution of the child networks. The expert networks, located at the leaf nodes, receive the input $\mathbf{x^{(m)}}$ to produce an estimate of the output. An example HME with binary branching is shown in Figure 1. This HME may be seen as a cascade of networks that works in a "bottom-up" fashion: the input is first presented to the experts that generate an output, then the output of the experts are combined by the second level gates, generating a new output. Finally the outputs of the second level gates are combined by the root gate to produce the appropriate result $\mathbf{y^{(n)}}$ (a vector $\mathbf{y}$ of n components where n is the number of outputs). The $\sum$ nodes in the tree represent the convex sum of the output of the child nodes which is computed as $\mathbf{y^{(n)}} = \sum_j g_j y_j^{(n)}$ where g_j is the output of the gate and $y_j^{(n)}$ is the output of the corresponding child node.

In the original model proposed by Jordan and Jacobs all the networks in the tree are linear (perceptrons). The expert network produces its output $\mathbf{y}$ as a generalized linear function of the input $\mathbf{x}$:

$$\mathbf{y} = f(U\mathbf{x}) \tag{1}$$

where U is a weight matrix and f is a fixed continuous non linear function. The vector $\mathbf{x}$ is assumed to include a fixed component with

value one to allow for an intercept term. For binary classification problems, $f(.)$ is the logistic function, in which case the expert outputs are interpreted as the log odds of "success" under a Bernoulli probability model. Other models (e.g., multi-way classification, counting, rate estimation and survival estimation) are handled by making other choices for $f(.)$.

The gating networks are also generalized linear functions. The i^{th} output of the gating network is the "softmax" function of the intermediate variable ψ_i [2, 23]:

$$g_i = \frac{e^{\psi_i}}{\sum_{j=1,...,l} e^{\psi_j}} \tag{2}$$

where l is the number of child nodes of the gating network, and the intermediate variable ψ_i is defined as:

$$\psi_i = \mathbf{v}_i^T \mathbf{x} \tag{3}$$

where $\mathbf{v}_i$ is a weight vector and ^{T} is the transpose operation. The g_is are positive and sum to one for each $\mathbf{x}$. They can be interpreted as providing a "soft" partitioning of the input space.

The output vector at each nonterminal node of the tree is the weighted sum of the output of the children below that nonterminal. For example, the output at the i^{th} nonterminal in the second layer of the two-level tree in Figure 1 is:

$$\mathbf{y}_i^{(n)} = \sum_j g_{i.j} \mathbf{y}_{i.j}^{(n)} \tag{4}$$

where $j = 1, ..., l$, l is the number of child nodes connected to the gate, $\mathbf{y}_{i.j}^{(n)}$ is the output of expert j which is a child of gate i, and $g_{i.j}$ is the jth output of the gate i. Note that since both the g's and the $\mathbf{y}$'s depend on the input $\mathbf{x}$, the output is a nonlinear function of the input.

The general HME model is very flexible. One may choose a function f (equation 1) for the gate and expert decision modules that is appropriate for the application. Moreover, since gates and experts depend only upon the input $\mathbf{x}$, one may choose either bottom-up or top-down processing whichever is appropriate for the problem. In our variation of the HME model we use a binary classification function at the gates. Also, we train and use our model top-down. The choice of this direction was made based upon the number of categories. As will be described in detail later, we have a collection of 119 categories in the dataset and moreover aim towards a scalable categorization procedure that can handle large classification schemes. The UMLS classification presently has more than $350,000$ concepts while the full OHSUMED

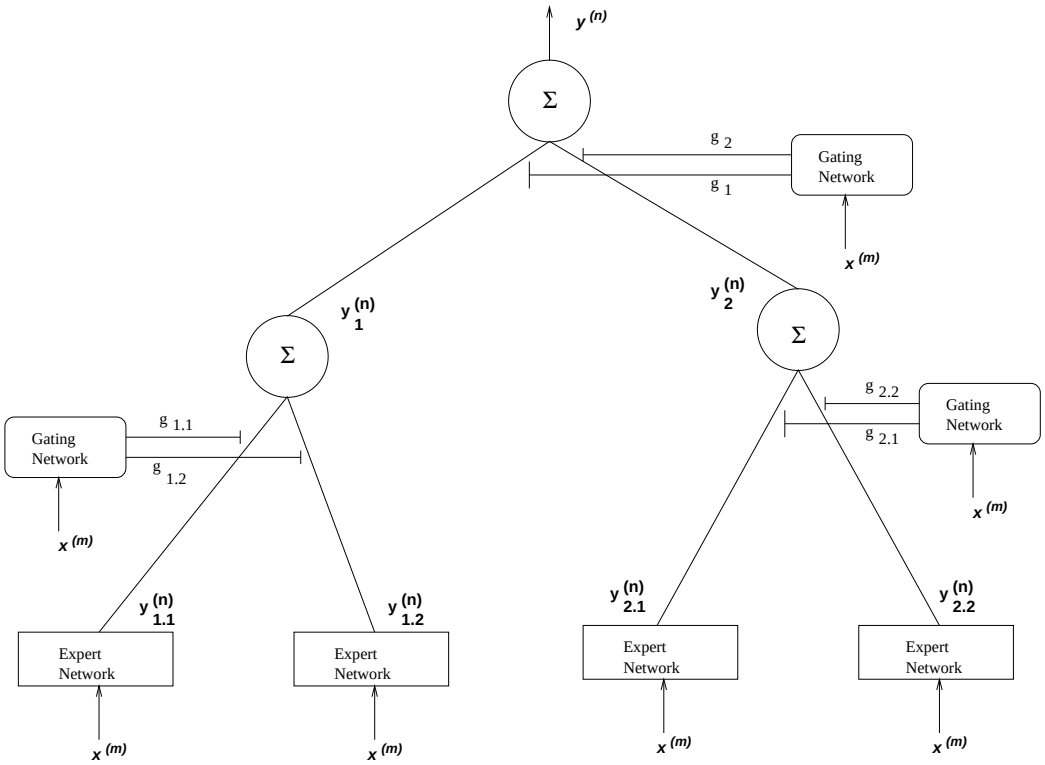

Figure 1. Hierarchical Mixture of Experts model

dataset has about 14,000 concepts. A bottom-up approach would be very inefficient given that there is an expert module for each category. The computational requirements are especially severe when the decision module at each expert node is a neural network. In contrast our top-down approach along with a binary classification at each gate restricts the number of expert networks to be activated for a given document. It may be observed that in studies using the bottom-up HME approach the number of categories was small as for example 26 for handwriting recognition and 52 for speech recognition [36]. In this study we choose the more efficient direction and postpone the exploration of bottom-up processing for future research.

In our model, given a binary function a gate is trained to yield a value of 1 if the example document is categorized with any of its descendent concepts. For example, in our domain the gate for the general concept of "Heart Disease" is trained to yield a 1 if the document is categorized by any of its specific categories such as "Coronary Thrombosis".

During testing, the categorization task starts at the root node and its gate decides whether the most general concept is present in the

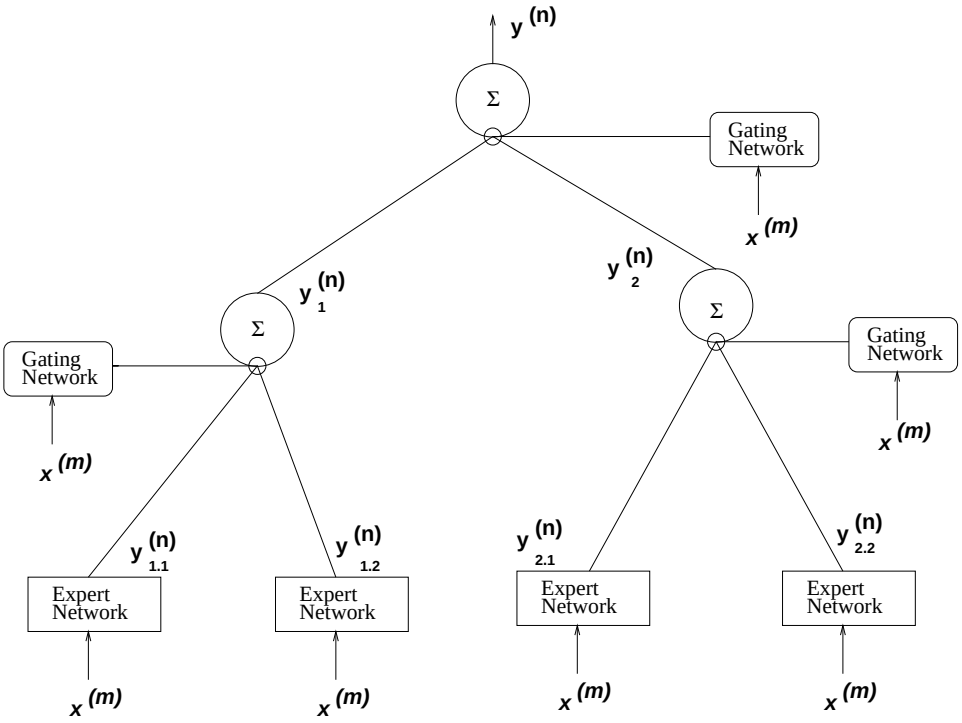

Figure 2. Modified Hierarchical Mixture of Experts model

document. If this is true, then all the second level nodes make their decisions and the process repeats until it reaches the leaf nodes. Observe that only the experts connected to gates that output the value 1 are activated thus reducing the response time during classification[1]. Figure 2 presents our modified HME classifier model. A key difference depicted is at the gating networks which use binary functions.

To give a statistical interpretation of our model we define $Z_1, \ldots, Z_j$ as the path of gates from the root node to the gate j that is the parent of an expert ε_k that assigns category k. Let $\mathbf{x}^{(m)}$ be the input features that represent a document, and $\mathbf{y}^{(n)}$ the output vector of the categories assigned to the document. The probabilistic interpretation of our hierarchical model is as follows:

[1] Hybrid neural trees have been proposed by D'Alché–Buc *et al.* [7] using a method called Trio-Learning to build a decision tree whose nodes are neural networks. Trio-learning uses a decision tree algorithm to partition the examples into positive and negative classes and recursively builds the hierarchy. Similar to CART this method performs a "hard split" of the set of examples at each level. Observe that our model uses a predefined hierarchical structure. Another difference is that we use a "soft split" of the data that allows overlapping clusters.

$$P(\mathbf{y^{(n)}}|\mathbf{x^{(m)}}) = \sum_{k=1}^{n} P(Z_1 = 1|\mathbf{x}^{(m)})P(Z_2 = 1|Z_1 = 1, \mathbf{x}^{(m)}) \ldots$$
$$P(Z_j = 1|Z_1 = 1, \ldots, Z_{j-1} = 1, \mathbf{x}^{(m)})P(\varepsilon_k = 1|\mathbf{x}^{(m)})$$

The hierarchical structure in a HME model is predefined[2] in our case by the hierarchy of the "Heart Disease" subset of the UMLS classification system. Moreover, the HME hierarchy is generally limited to the classes that appear in the training set. Given our training set (described later) we are able to use only 103 concepts of the 119 in the "Heart Disease" subset.

There are several alternatives for training a HME model. Jordan and Jacobs [13] and Waterhouse [36] use a method based on expectation maximization. They assume that the classification follows a multinomial model, which implies that an object can be assigned to one and only one of the multiple categories available for classification. This is a 1-of-K classification task which may be viewed as a competition problem. In contrast, we are interested in a k-of-K (multi-way) classification problem which is equivalent to k independent 1-of-2 classifications[23, 29]. To allow for multi-way classification we use backpropagation neural networks in both gates and experts and use a gradient descent method for training. The gates are trained to recognize whether or not any of the categories of their descendants is present in the document. The experts are trained to recognize the presence or absence of particular categories.

The backpropagation networks that we use have three layers (see Figure 3). We have tested several configurations and these results will be discussed in detail later. In general, our neural networks have m nodes in the input layer corresponding to the set of m features selected for each expert (or gate), the middle layer has n nodes, and the output layer is a single node. In this case the sigma nodes are responsible for combining the outputs of the gate i and expert j to obtain the kth component $y_k = g_i y_{i.j}$ (where $k = 1..n$) of the output $Y_i^{(n)}$ at each intermediate node of the HME. Observe that the general model allows for experts to have an output vector of size n. Our implicit assumption for this paper is that each category assignment is independent.

Given an appropriate set of features and a training set of manually categorized documents, the backpropagation network learns to make the appropriate decisions. Observe that for experts and gates the set of positive examples is different. The set of positive examples for the

[2] It should be noted that Waterhouse has proposed a HME model that dynamically generates the hierarchical structure [36].

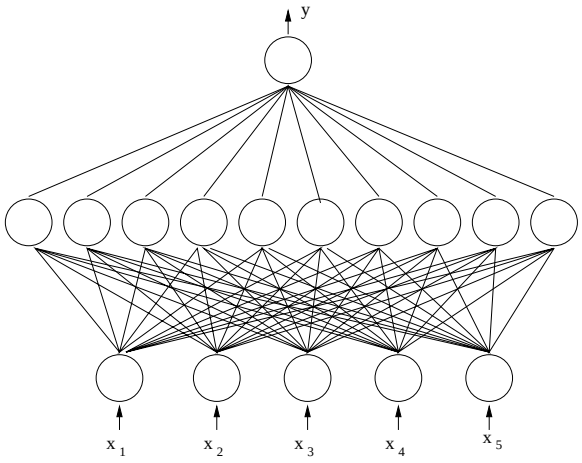

Figure 3. Example of a backpropagation network with 5 inputs

experts is a subset of the positive examples of any ancestor gate. As a consequence, two identical neural networks trained with these different subsets learn different probabilistic functions. The backpropagation neural network as an expert node learns to use the input to estimate the desired output value (category), while as a gate it computes the confidence value of the combined outputs of its children.

3. Implementation of the HME

We want to build a classifier that is able to effectively use the structured knowledge contained in the UMLS Metathesaurus [26]. In particular we are interested in the hierarchical relationships which link general concepts to the more specific ones. The 1999 UMLS Metathesaurus contains about $350,000$ concepts collected by combining 79 vocabularies for the health sciences. In this study we limit ourselves to MeSH (Medical Subject Headings)[3] which is one of the 79 vocabularies. Since documents in the OHSUMED test collection are a subset of MEDLINE, they have been manually categorized with MeSH terms. Each MEDLINE document is assigned between 8 and 10 MeSH concepts. Thus our categorization task is a multi-way classification problem. In our model the gates represent the general concepts of this hierarchy.

Interestingly, the manual assignment of a high level MeSH category is not automatically determined by the assignment of its lower level

[3] Observe that we could have used the MeSH hierarchy directly. We decided to use the UMLS hierarchy because it provides a conceptual mapping that has been extended to many areas of the health sciences.

categories. That is, the fact that a document is assigned the category "angina unstable" does not automatically grant it the assignment of any of the ancestors in the tree ("Heart diseases", "Myocardial Ischemia", "Coronary Diseases", or "Angina Pectoris"). In fact the manual assignment of such high level categories is usually done when the MEDLINE document is about the topic at the associated level of generality or abstraction. Therefore in our model, each nonterminal node is represented by two networks. The first is the expert network for the node's category while the second is a gating network representing the general concept at that level of the classification scheme. Thus at the "Heart Diseases" node there is a gate that learns to recognize the general *concept* (representing all the documents that are about any of its descendants), and an expert network that learns to assign this specific category "Heart Diseases". Note that from this point on unless explicitly specified, we mean both categories and concepts when we use the word 'category'.

For the purpose of comparing results with other studies we will show the results obtained using only the MeSH subtree of "Heart Diseases". (Figure 4 shows a part of this hierarchy). However, our method especially given its top-down processing, is general and can be applied to the whole set or to any other subset of the UMLS.

4. Feature Selection

In text categorization the set of possible input features consists of all the different words that appear in a collection of documents. This is usually a large set since even small text collections could have hundreds of thousands of features. Reduction of the set of features to train the neural networks is necessary because the performance of the network and the cost of classification are sensitive to the size and quality of the input features used to train the network [41]. A first step towards reducing the size of the feature set is the elimination of stop words, and the use of stemming algorithms. Even after that is done the set of features is typically too large to be useful for training a neural network.

Two broad approaches for feature selection have been presented in the literature: the wrapper approach, and the filter approach [12]. The wrapper approach attempts to identify the best feature subset to use with a particular algorithm. For example, for a neural network the wrapper approach selects an initial subset and measures the performance of the network; then it generates an "improved set of features" and measures the performance of the network. This process is repeated until it reaches a termination condition (either a minimal value of performance or a number of iterations). The filter approach, which

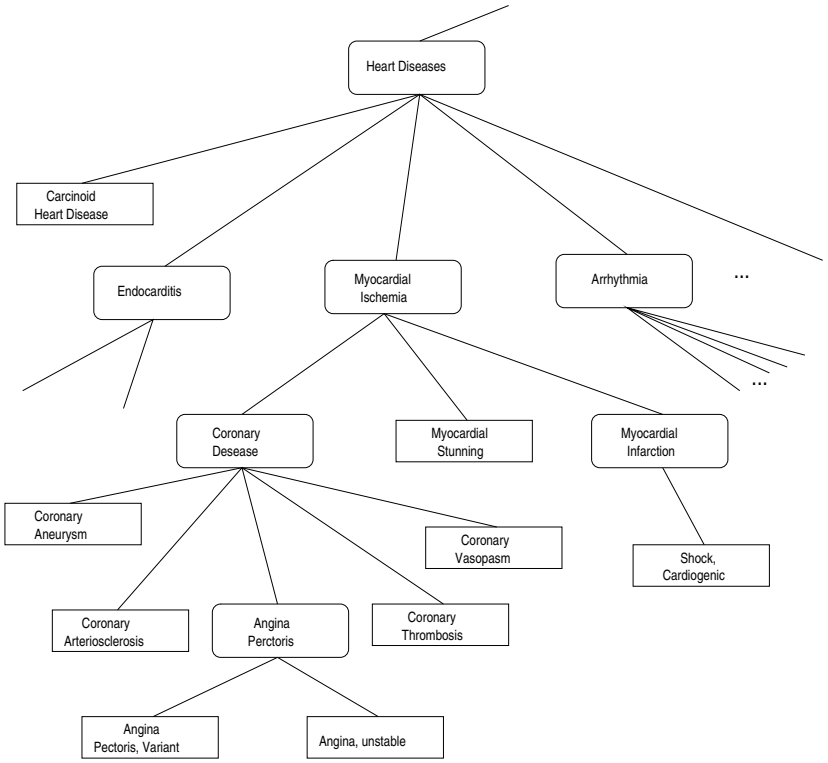

Figure 4. A part of the UMLS hierarchy for the heart diseases subtree.

is more commonly used in text categorization, attempts to assess the merits of the feature set from the data alone. The filtering approach selects a set of features using a preprocessing step, based on the training data. In this paper we use the filter approach, but we plan to explore the wrapper approach in future research. We select three methods that have been used in previous works: correlation coefficient, mutual information, and odds ratio. During feature selection we first delete all instances of 571 stop words from the MEDLINE records, and then use Porter's algorithm to stem the remaining words. We eliminate those stems that occur in less than 5 documents in the training collection. Since feature selection is done for each category, based on its zone (explained later) we also remove stems that occur in less than 5% of the positive example documents. We then rank the remaining stems by the feature selection measure and select a pre-defined number of top ranked stems as the feature set.

4.1. Correlation Coefficient

Correlation coefficient C is a feature selection measure proposed by Ng *et al.* [27] and is defined as:

$$C(w,c) = \frac{(N_{r+}N_{n-} - N_{r-}N_{n+})\sqrt{N}}{\sqrt{(N_{r+}+N_{r-})(N_{n+}+N_{n-})(N_{r+}+N_{n+})(N_{r-}+N_{n-})}} \tag{5}$$

where $N_{r+}(N_{r-})$ is the number of positive examples of category c in which feature w occurs(does not occur), and $N_{n+}(N_{n-})$ is the number of negative examples of category c in which feature w occurs(does not occur). This measure is derived from the χ^2 measure presented by Schütze *et al.* [32], where $C^2 = \chi^2$. The correlation coefficient can be interpreted as a "one-side" χ^2 measurement. The χ^2 measure has been reported as a good measure for text categorization by Yang and Petersen [43]. The correlation coefficient promotes features that have high frequency in the relevant examples but are rare in the non relevant documents. When features are ranked by this method, the positive values correspond to features that indicate presence of the category while the negative values indicate absence of the category. In contrast, the χ^2 ranks features higher if they more strongly indicate the presence or the absence of a category. That is, more ambiguous features are ranked lower. We compared the χ^2 and correlation coefficient for feature selection using neural networks with the same architecture. We found that the neural networks trained with features selected using correlation coefficient outperformed those trained using χ^2 in 78 out of 103 categories. This confirms similar results reported by Ng et al. [27]. Note that Yang and Pedersen [43] use an average of the χ^2 value across categories to measure the goodness of a term in a global sense, while we use it for local (category-level) feature selection.

In contrast with mutual information, both χ^2 and correlation coefficient produce normalized values because they are based on the χ^2 statistic. However, the normalization does not hold for low populated cells in the contingency table. This makes the scores of χ^2 and correlation coefficient for low frequency terms unreliable. This is one reason for removing rare features as described before.

4.2. Mutual Information

Mutual information is a measure that has been used in text categorization by several researchers [32, 43]. This method is based on the mutual information concept developed in information theory. For a feature w and a category c it is defined as:

$$I(w,c) = \log \frac{P(w \wedge c)}{P(w) \times P(c)} \tag{6}$$

where $P(w)$ is the probability of the term w occurring in the whole collection, $P(c)$ is the probability of the category c occurring in the whole collection, and $P(w \wedge c)$ is their joint probability.

Yang and Pedersen [43] used mutual information[4] to measure the goodness of a term in a global feature selection approach by combining the category specific scores of a term in two ways:

$$I_{avg}(w) = \sum_{i=1}^{n} P(c_i) I(w, c_i) \tag{7}$$

$$I_{max}(w) = \max_{i=1}^{n} \{I(w, c_i)\} \tag{8}$$

where n is the number of categories.

In contrast, we use feature selection to evaluate the goodness of a term with respect to individual categories. In other words, we do not average the values of mutual information over multiple categories. This variation may be sufficient to produce the different results that we obtain (described later). Yang and Pedersen also point out that the score produced by mutual information is strongly influenced by the marginal probabilities of terms. This is evident from the following equivalent formula:

$$I(w,c) = \log P(w|c) - \log P(w) \tag{9}$$

For terms with equal conditional probability $P(w|c)$, rare terms will have higher scores than common terms. This implies that the scores of terms with extremely different frequencies might still not be comparable. Our frequency threshold described earlier, compensates for this effect.

4.3. Odds Ratio

Odds ratio was proposed originally by van Rijsbergen *et al.* [35] for selecting terms for relevance feedback. Odds ratio is used for the binary-valued class problem where the goal is to make a good prediction for one of the class values [35]. It is based on the idea that the distribution of features on the relevant documents is different from the distribution of features on the non-relevant documents. It has been recently used

[4] There is some confusion with the term "mutual information". For instance, it has been used by other researchers [22] to refer to the measure that Yang and Pedersen [43] present as "information gain".

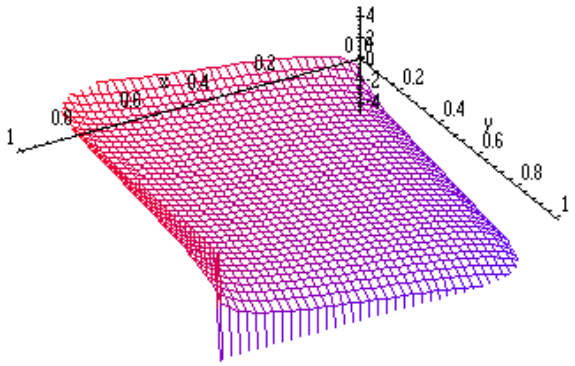

Figure 5. Graph of Odds Ratio. The x axis represents $P(w|pos)$, while the y axis represents the $P(w|neg)$.

by Mladenić [24] for selecting terms in text categorization. The odds ratio of a feature w, given the set of positive examples pos and negative examples neg for a category c, is defined as follows:

$$OddsRatio(w, c) = \log \frac{P(w|pos)(1 - P(w|neg))}{(1 - P(w|pos))P(w|neg)} \tag{10}$$

Observe that this formula can also be interpreted as the sum of the logarithm of the ratios of the distribution of the feature on the relevant documents ($\log \frac{P(w|pos)}{(1-P(w|pos))}$) and on the non-relevant documents ($\log \frac{(1-P(w|neg))}{P(w|neg)}$). If a document appears in more than half of the relevant documents the logarithm of the ratio on the relevant documents is positive. In contrast a feature is penalized if it appears in more than half of the non-relevant documents. In other words, a feature that appears frequently in the relevant documents and infrequently in the non relevant documents will have a high score. Figure 5 shows a graph of the odds ratio. The function presents singularity points when $P(w|pos) = 1$ or when $P(w|neg) = 0$ (we map this case to the highest positive value). Also the logarithm is not defined when $P(w|pos) = 0$ or when $P(w|neg) = 1$ (we map this case to the smallest negative value).

Mladenić [24] report that odds ratio was the most successful feature selection method for a hierarchical Bayesian classifier compared to mutual information, cross entropy, information gain, and weight of evidence.

In this study we select features for both expert and gating networks using correlation coefficient, mutual information and odds ratio methods.

5. Training Set Selection

A supervised learning algorithm requires the use of a training set in which each element has already been correctly categorized. One would expect that the availability of a large training set (such as OHSUMED) will be beneficial for training the algorithm. In practice this does not seem to be the case. The problem occurs when there is also a large collection of categories with each assigned to a relatively small number of documents. This then creates a situation in which each category has a small number of positive examples and an overwhelming number of negative examples. When a machine learning algorithm is trained to learn the assignment function with such an unbalanced training set, the algorithm will learn that the best decision is to not assign the category. The overwhelming amount of negative examples hides the assignment function. To overcome this problem an appropriate set of training examples must be selected. We call this training subset the "category zone". This notion of category zone is similar to the local regions described in Wiener *et al.* [39] , and Ng *et al.* [27] but is inspired by the query zone proposed by Singhal *et al.* [34] for text routing. Their "query zoning" is based on the observation that in a large collection a query will have a set of documents that constitutes its domain. Non-relevant documents that are outside the domain are easy to identify, but it is more difficult to differentiate between relevant and non-relevant documents within the query domain. Singhal *et al.* [34] define a procedure that tries to approximate the domain of the query and then they use this domain to train their routing method. We suggest that in text categorization, each category also has its own domain. It will be easier to train a learning algorithm with those documents from the category domain and also potentially achieve better categorization performance. We explore two different methods for building the category zone. The first method creates the category zone using a method similar to that presented by Singhal *et al.* [34]. This first category zone that we call centroid-based is created as follows:

1. Take all the positive examples for a category and obtain their centroid.

2. Using this centroid as a query perform retrieval and obtain the top $10,000$ documents. This subset will contain most if not all of the

positive examples and many negative examples that are at least "closely related" to the domain of the category.

3. Obtain the category zone by adding any unretrieved positive examples to the set obtained in the previous step.

This method creates category zones that have at least 10,000 documents and the size increases for categories that have positive examples outside the retrieved set.

The second method for creating the category zone uses a Knn approach in which the category zone consists of the set of K nearest neighbors for each positive example of the category. This method will produce variable sized category zones. We explored several values of K (10, 50, 100 and 200). Our main concern with this method was to obtain a training set large enough to train a neural network without overfitting.

6. Experimental Collection

We use the OHSUMED collection created by Hersh and his collaborators [9]. This collection has 348,543 records from the MEDLINE collection from 1987 to 1991. Each record from this collection has several fields (see Figure 6). We use the following: title (.T), and abstract (.W). In the training set we also use the MeSH (.M) field which represents the manual categorization decisions for the MEDLINE documents. We selected the 233,455 records that have titles, abstracts and MeSH categories (the remaining do not have abstracts). The first four years of data dated 1987 through 1990 (183,229 records) are used for training, and the year 1991 (50,216 records) is used for testing. This corresponds to the same split as used by Lewis *et al.* [19]. We also use the 119 categories from the Heart Disease subtree of the Cardiovascular Diseases tree structure of the UMLS[5]. However, only 103 of these 119 categories have positive examples in the training set. Thus we limit our experiments to these 103 categories. We further divide this set of 103 categories into three sets:

– High frequency categories (HD-49): This includes all categories with at least 75 examples in the training set. This set contains 49 categories (which is the same as the set of high frequency categories used by Lewis *et al.*).

[5] We use the 1994 version of the UMLS.

```
.I 55402
.S
Heart Lung 8801; 16(5):584-9
.M
Adult; AIDS-Related Complex/*DI; Cardiac Tamponade/DI/ET;
Case Report; Electrocardiography; Endocarditis, Subacute
Bacterial/DI/ET; Female; Heart Diseases/DI/*ET/RA; Heart
Failure, Congestive/DI/ET; Heart Valve Diseases/DI/ET;
Human; Male; Myocardial Diseases/DI/ET; Support, Non-U.S.
Gov't.
.T
The Miami vices in the CCU. Part II. Cardiac manifestations
of AIDS.
.W
Cardiac manifestations of AIDS probably occur more frequently
than is appreciated --despite autopsy reports indicating that
more than 50\% of deceased AIDS patients had myocarditis.
A high index of suspicion and the echocardiogram will help in
revealing the true incidence of cardiac involvement in AIDS.
.A
Valle BK; Lemberg L.
```

Figure 6. Example record from OHSUMED

- Medium frequency categories (HD-28): This set includes all categories with frequencies between 15 and 74 in the training set. This set contains 28 categories (this is equivalent to the second set of categories used by Lewis *et al.*).
- Low frequency categories (HD-26): This set includes all categories with frequencies between 1 and 14 in the training set. This set contains 26 categories.

We report results on these three subsets as well as on the complete set of categories (HD-119)[6]. The first two subsets allow us to analyze performance separately for different levels of positive evidence and also allow us to compare results with other published research with the same collection [19].

The 119 "Heart Diseases" categories form a 5 level tree where the first level corresponds to the root node and the fifth level has only leaf

[6] We label this set HD-119 in order to stay consistent with the labeling in [42] even though there are actually only 103 categories with positive examples in the training set.

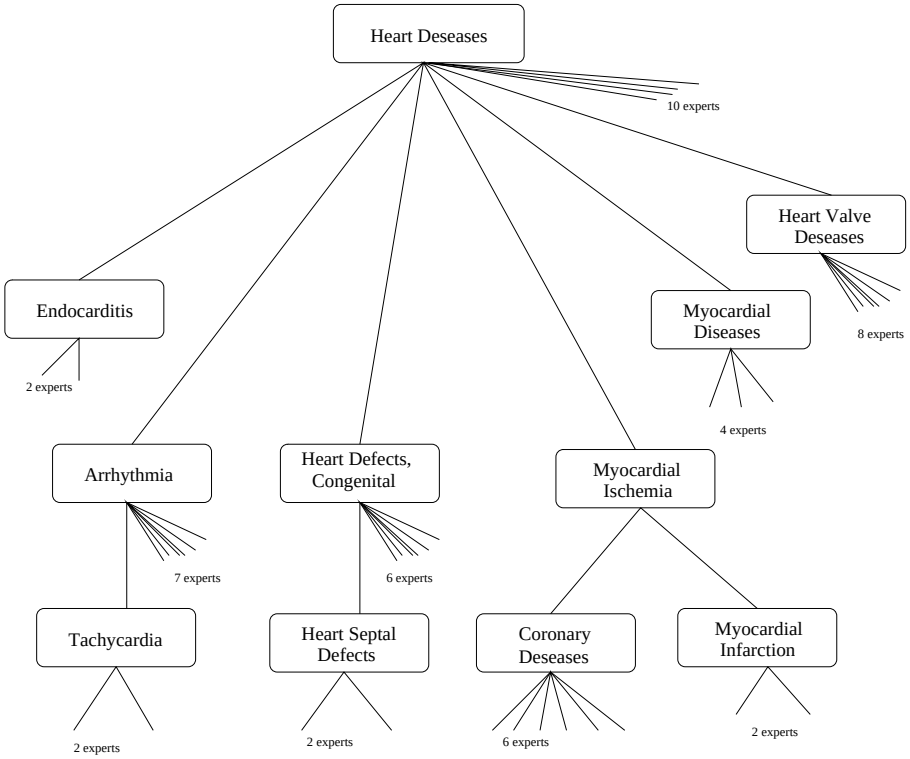

Figure 7. Tree for the 119 categories of the heart diseases sub-tree

nodes. The number of gates in each level starting from the root is 1, 11, 9, and 3 (see Figure 7).

7. Evaluation Measures

In order to evaluate a binary decision task we first define a contingency matrix representing the possible outcomes of the classification as shown in Table I. Several measures in the IR and AI communities have been defined based on this contingency table. Table II shows the formulas of these measures and the corresponding names used in each community.

In IR measures that combine recall and precision have been defined: break-even point (BEP), and F_β measure. BEP was proposed by Lewis [18] and is defined as the point at which recall equals precision. van Rijsbergen's F_β measure [35] combines recall and precision into a single score:

$$F_\beta = \frac{(\beta^2+1)P \times R}{\beta^2 P + R} = \frac{(\beta^2+1)a}{(\beta^2+1)a + b + \beta^2 c} \qquad (11)$$

Table I. Contingency table for binary classification

	Class Positive ($C+$)	Class Negative ($C-$)
Assigned positive ($A+$)	a (True Positives)	b (False Positives)
Assigned negative ($A-$)	c (False Negatives)	d (True Negatives)

Table II. Efficiency measures for binary classification defined in the Information Retrieval (IR) and Artificial Intelligence (AI) communities

IR	**AI**	
recall (R)	Sensitivity	$\frac{a}{a+c}$
precision (P)	Predictive_value(+)	$\frac{a}{a+b}$
fallout	Predictive_value(-)	$\frac{b}{b+d}$
	Accuracy	$\frac{a+d}{a+b+c+d}$
	Specificity	$\frac{d}{b+d}$
error_rate		$\frac{b+c}{a+b+c+d}$

F_0 is the same as precision, F_∞ is the same as recall. Intermediate values between 0 and ∞ are different weights assigned to recall, and precision. The most common values assigned to β are 0.5 (recall half as important as precision), 1.0 (recall and precision equally important) and 2.0 (recall twice as important as precision). If a, b and c are all 0, F_β is defined as 1 (this occurs when a classifier assigns no documents to the category and there are no related documents in the collection).

None of the measures are perfect or appropriate for every problem. For example, recall (sensitivity), if used alone might show deceiving results, i.e. a system that assigns the category to every document will show perfect recall (1.0). Accuracy, on the other hand works well if the number of positive and negative examples are balanced, but in extreme conditions it too might be deceiving. If the number of negative examples is overwhelming compared to the positive examples then a system that assigns no documents to the category will obtain an accuracy value close to 1.

As pointed out by Schapire *et al.* [31], BEP also shows some problems. Usually the value of the BEP has to be interpolated. If the values of recall and precision are too far then BEP will show values that are not achievable by the system. Also the point where recall equals precision is not informative and not necessarily desirable from the user's perspective.

van Rijsbergen's F measure is the best suited measure, but still has the drawback that it might be difficult for the user to define the relative importance of recall and precision. We report F_1 values because it allows us to compare results with other researchers who have used the same dataset [16, 19, 42]. In general the F_1 performance is reported as an average value. There are two ways of computing this average: macro-average, and micro-average. With macro-average the F_1 value is computed for each category and these are averaged to get the final macro-averaged F_1. With micro-average we first obtain the global values for the true positive, true negative, false positive, and false negative decisions and then compute the micro-averaged F_1 value using the micro-recall and micro-precision (computed with these global values). The results reported in this paper are macro-averaged F_1. This allows us to compare our results with those of other researchers working with the OHSUMED dataset.

8. Experiments

There are two main questions that we address in this research (1) Does a hierarchical classifier built on the HME model improve performance when compared to a flat classifier? (2) How does our hierarchical method compare with other text categorization approaches? With these two research questions in mind we present a series of experiments using the OHSUMED collection.

8.1. Baselines

Our first baseline represents a classical Rocchio classifier which is described in the next section. Our second baseline is a flat neural network classifier. Comparing the performance of the HME classifier against the flat classifier will allow us to answer our first research question. Comparing the HME method with a Rocchio classifier as well as with other published results will allow us to answer our second research question. Thus we have implemented a Rocchio classifier, a HME classifier, and a flat neural network classifier which are detailed next.

8.2. Rocchio Classifier

Rocchio's algorithm was developed in the mid 1960's to improve queries using relevance feedback. It has proven to be one of the most successful feedback algorithms. Rocchio [28] showed that the *optimal query* vector is the difference vector of the centroid vectors for the relevant and the non-relevant documents. Salton and Buckley [4] included the original query (Q_{orig}) to preserve the focus of the query, and added coefficients (α, β and γ) to control the contribution of each component. The mathematical formulation of this version is:

$$\vec{Q}_{new} = \alpha\vec{Q}_{orig} + \beta\frac{1}{R}\sum_{d\in rel}\vec{d} - \gamma\frac{1}{N-R}\sum_{d\notin rel}\vec{d} \qquad (12)$$

where $\vec{d}$ is the weighted document vector, $R = |Rel|$ is the number of relevant documents, and N is the total number of documents. Any negative components of the final vector $\vec{Q}_{new}$ are set to zero. Several techniques have been proposed to improve the effectiveness of Rocchio's method: better weighting schemes [33], query zoning [34], and dynamic feedback optimization [4].

As pointed out by Schapire *et al.* [31] most of the studies that use Rocchio as a baseline have constructed a weak version of the classifier [16, 19, 42, 44]. They also show that a properly optimized Rocchio's algorithm could achieve quite competitive performance. We have noticed that Rocchio's classifiers benefit from an optimal feature selection step. To make a fair comparison between the neural networks and the Rocchio classifiers we use the set of features selected using correlation coefficient and the same category zones used to train the neural network classifiers for each category. Observe that this is an important difference with respect to previously published research that use Rocchio classifiers (in all these studies the vector is computed over the whole set of features). Since we use feature selection measures that select features indicative of presence of the category, each classifier has its centroid vector defined in a different subspace (the sub-space of the selected features) generated from the category zone.

We build a Rocchio classifier by presenting training examples from the category zone and computing the weights of the classifier using Rocchio's formula. We then rank the full training collection according to the similarity with this classifier vector. A threshold (τ) on the similarity value that maximizes the F_1 measure (described in the evaluation measures section) is selected. The *optimal* Rocchio classifier for a category is then a weighted vector of selected features along with a similarity threshold.

During the evaluation phase we compute similarity between the optimal Rocchio classifier vector and the test document vectors and assign the class to all documents above the threshold τ.

8.3. Hierarchical Mixture of Experts

Our HME approach is represented in Figure 2. First a zone of domain documents is identified for each category as explained before. Next feature selection is applied within each category zone to extract the "best" set of features. We tested all three feature selection methods in our experiments. Then for each expert network a backpropagation neural network is trained using the corresponding category zone and the selected set of features. Similarly, each gating network is also a backpropagation network. However a gate's training subset is the combined category zones of its descendants in the classification hierarchy. Feature selection for the gate is then performed on its combined subset. This strategy of combining zones from descendent nodes for a gate is reasonable if we consider the fact that gates represent hierarchical concepts and not particular categories as described before in section 3.

The input feature vectors for documents are weighted using $tf \times idf$ weights where tf is the frequency of the term in the document, and idf is the inverse document frequency defined as:

$$idf = \log \frac{N}{n} \tag{13}$$

where N is the total number of documents in the training collection, and n is the number of documents that contain the term in the training collection.

Experts and gates are trained independently using the following parameters: learning rate $= 0.5$, error tolerance $= 0.01$, maximum number of epochs $= 1,000$. These parameter values are fixed for all our experiments. The training of each network takes between 15 to 30 minutes for an expert (depending on the number of examples), and around 60 to 90 minutes for a gating network using a HP-700 workstation. Using 15 workstations and a dynamic scheduling program specifically designed for this task we trained the 103 experts and the 21 gating networks in about 8 hours.

Once experts and gates have been trained individually we assemble them according to the UMLS hierarchical structure. Since the output of each network is a real value between 0 and 1, we need to transform each output value into a binary decision. This step is called thresholding. We do this by selecting thresholds that optimize the F_1 values for the categories. We use the complete training set to select the optimal

thresholds. Since we are working with a modular hierarchical structure we have several choices to perform thresholding. Our approach is to make a binary decision in each of the gates and then optimize the threshold on the experts using only those examples that reach the leaf nodes.

Observe that computing the optimal thresholds for binary decisions at the gates and the experts is a multidimensional optimization problem. We decided to optimize the gates by grouping them into levels and finding the value of the threshold at each level that maximizes the average F_1 value for all the experts. Each expert's threshold is then optimized to maximize the F_1 value of the examples in the training set that reach the expert. In order to constrain the potentially explosive combination of parameters we decide to fix the thresholding for the gates across all our experiments. For this purpose we conducted a preliminary experiment in which we search the best combination of thresholds per level varying each threshold on fixed values and computing performance on the whole training set. The optimal thresholds were set to 0.01, 0.005, 0.01 and 0.01 for levels 1(root), 2, 3 and 4 respectively. These values were obtained by selecting the best results over $1,764$ threshold combinations (0.005, 0.01, 0.05, and 0.10 for level 1, 0.005, 0.01, 0.05, 0.10, 0.15, 0.20, ... 0.95 for levels 2, 3 and 4)[7]. This experiment was run using correlation coefficient for feature selection and a standard configuration of 25 input nodes and 50 hidden nodes. The best threshold are fixed across all runs.

The test set is processed using the trained networks assembled hierarchically with the established thresholds for each level of gates and each expert network.

8.4. Flat Neural Network Classifier

In order to assess the advantage gained by exploiting the hierarchical structure of the classification scheme, we built a flat neural network classifier. We decided to build a flat modular classifier that is implemented as a set of 103 individual expert networks. This is similar to our Rocchio model where the training phase results in a set of 103 classifier vectors. In this model the experts are trained independently using the optimal feature set and the category zone for each individual category. The thresholding step is performed by optimizing the F_1 value of each expert using the entire training set. These are the values that we report in the next section for the flat neural network classifier. Observe also

[7] Since we only have two gates in level 4 we set their optimal values to the same values of the gates in level 3. This gives us $4 \times 21 \times 21 = 1,764$ possible combinations for the thresholds in the gates.

that the use of this model allows us to assess the contribution of adding the hierarchical structure, i.e., our first research question.

9. Results

As stated before, we report results on the "Heart Diseases" sub-tree. We present results on this set of 103 categories (HD-119) and on three frequency-based subsets of categories HD-49, HD-28 and HD-26 as defined in section 6.

9.1. Effect of feature selection and the Neural Network Architecture

It may be observed that network architecture and feature selection methods must be studied in combination. In fact, at a basic level feature selection is one of the factors that define the configuration of the network. Given the many complex combinations in terms of feature selection methods and numbers of nodes in the different layers for both the expert and gating networks we approach the problem in stages. We follow a top-down approach that optimizes the gates and then the experts.

First we focused our attention on the gating networks. We used experts with 25 input features and 50 nodes in the hidden layers. We then explored 5, 10, 25, 50, 100, and 150 input features for the gating networks with hidden layer that had twice the number of input nodes. We also tried all three different feature selection methods (Mutual Information, Odds Ratio and Correlation Coefficient). This experiment was done only on the "high frequency categories" (HD-49) because they allow appropriate training of the neural networks and also because the variance between different training runs is smaller than the variance for lower frequency categories.

Interestingly the differences between the three feature selection methods on the gating networks are not significant. Thus we only report results on the 18 different combinations obtained using correlation coefficient in the gates. Table III shows an increase in performance between 5 and 25 features and a slight decrease for networks with larger number of input nodes. It is possible for this slight decrease to be caused by the limit in the number of iterations $(1,000)$ that a network was allowed to run during training. Usually a larger network needs more iterations on the training set to converge to an optimal value. Observe that all the three feature selection methods show no significant differences either for the gates or for the expert networks. This was somewhat surprising

since Yang and Pedersen [43] reported that mutual information does not perform well compared to other methods. As noted before, instead of averaging values across multiple classes to find the merit of the features from a global perspective, we use mutual information for local feature selection. We also discard rare terms that will in general be ranked very high by mutual information.

We further explored feature selection using mutual information by running our experiments without discarding rare terms and selecting the top 25 features. The average precision for the HD-49 subset was 0.05 and 0.20 for the flat neural network and the HME model respectively. This is significantly lower than the performance obtained when we discard low frequency terms and shows conclusively that mutual information alone is not a good feature selection measure unless we address its major weakness and discard low frequency terms. This might also be addressed by selecting a larger number of input features. However, this will go against our goal of reducing the number of input features to improve training and processing time for the neural networks.

After optimizing the gates we address the number of input nodes for the expert networks by exploring them individually, i.e., independent of the hierarchical structure. We tested expert networks with 5, 10, 25, 50, 100, and 150 input features. In each case we used twice the number of input nodes for the hidden layer and three feature selection methods. The best result was obtained using 25 input features with 50 nodes in the hidden layer.

Having determined the optimal number of inputs, next we explore the effect of the size of the hidden layer for our networks. (Note that so far we have only explored the simple strategy of having twice the input nodes in the middle layer.) Table IV shows the variations in performance with different sizes of the middle layer. In this case all networks have 25 inputs and a single output node. The best performance is obtained with expert networks that have 6 nodes in the hidden layer. The difference between 6 and 10 nodes is relatively small. We ran similar experiments on the gating networks varying the size of the middle layer (6, 10, 25, 50) and found that the best size of the hidden layer was 25 (Table V). However, the difference between these runs is very small. Observe also that the flat neural networks have their best performance when the number of nodes in the hidden layer is 6. This is not surprising since a smaller hidden layer tends to produce a better generalization of each category.

In the following sections we present results using neural networks with 25 inputs, 6 hidden nodes and 1 output for the experts and 25 inputs, 25 hidden nodes and 1 output for the gating network.

Table III. Effect of the number of input nodes to the gating networks and the feature selection methods for the expert networks (the feature selection method for the gating networks is correlation coefficient). All expert networks have 25 input features selected by the indicated feature selection method, 50 nodes in the middle layer, and 1 output. The last row shows performance of the flat classifier. (Performance is measured in macro-averaged F_1 on the HD-49 set)

Gating Networks	Expert Networks		
# of inputs	Corr. Coef.	Odds Ratio	Mutual Inf.
5	0.4455	0.4531	0.4589
10	0.4604	0.4695	0.4712
25	**0.4984**	**0.4961**	**0.4956**
50	0.4894	0.4903	0.4900
100	0.4894	0.4929	0.4840
150	0.4827	0.4890	0.4850
Flat	0.4449	0.4488	0.4548

Table IV. Effect of the number of hidden nodes on the expert networks. All expert networks have 25 input features selected with the correlation coefficient feature selection method. (Performance is measured by macro-averaged F_1 on the HD-49 set)

# of hidden nodes	Flat NN	HME
6	0.5033	0.5241
10	0.4807	0.4975
25	0.4320	0.4824
50	0.4479	0.4867

Table V. Effect of the number of hidden nodes on the gating networks. All expert networks have 25 input features and 6 hidden nodes. The gates have 25 inputs selected with Correlation coefficient(Performance is measured by macro-averaged F_1 on the HD-49 set)

# of hidden nodes	HME
6	0.5219
10	0.5202
25	0.5241
50	0.5158

9.2. Comparing the Category Zone Methods

As mentioned before, we explore two different types of category zones: The centroid-based category zone and the Knn-based category zone. We compare these two zoning strategies with respect to their zone sizes as well as classifier performance.

The centroid-based category zone generates zones that have at least $10,000$ examples. For our training set of 103 categories we found that this type of category zone has an average size of $10,027$ with a maximum of $10,778$ while 75% of the zones are below $10,010$. The Knn-based category zone generate zones with sizes proportional to the number of positive examples in the category. We found that "compact" categories have in general small category zones. We explore different values of K (10, 50, 100, and 200). Small values of K are problematic for low frequency categories because they tend to generate a very small training set that the neural network overfits easily. Thus we settled for K=200 because it produces category zones large enough for the rare categories as well as zones of reasonable size for the more frequent categories. For our 103 categories we found that the average size of the Knn-based category zone is $6,098$ examples with a maximum of $35,917$, and a minimum of 200. 75% of the category zones generated by this method are below $6,814$. As mentioned before the category zone for a gate is the union of the individual category zones of its descendants in the hierarchy.

To measure the impact of each zoning method we trained both gating and expert networks with the documents of the corresponding zones. The categorization results on the test set are shown in Table VI. There are small differences in performance across zones for classifiers in the high frequencies (HD-49) set and for those classifiers in the medium frequencies (HD-28) set. However, these differences are not statistically significant.

The low frequency categories (HD-26) show a statistically significant difference in favor of the centroid-based zones for both flat and hierarchical classifiers. A detailed analysis showed that the high value of this difference is due in part to the contribution of some categories that have one or zero examples in the test set. For these categories the function becomes more of a hit or miss function[8]. Since these category zone training sets have at least $10,000$ examples the classifier learns to reject most of the documents and in consequence it gets an F_1 value of 1.0 for most of them. In contrast, the Knn-based zones for low frequency categories generate a smaller zone and the neural networks trained with them tend to assign at least a few documents. The classifiers trained with centroid-based zones outperform the Knn-based classifiers on 13 categories, while the classifiers trained on Knn-based zones outperform the centroid-based classifiers only on 5 categories (in the remaining 8 categories there is no difference between them).

On the whole set HD-119 the classifiers trained with centroid-based zones outperform those trained with the Knn-based zones. However this difference is statistically significant only for the HME classifiers. Given our results with these two zoning methods we suggest that centroid-based category zones are the most appropriate for training hierarchical classifiers that span categories of varying frequencies. The same conclusion may be made for flat classifiers but with somewhat reduced confidence.

9.3. Comparing HME, Flat NN, and Optimized Rocchio

Table VII shows the performance of our flat neural network, the HME model and the optimized Rocchio classifier, all trained using the centroid-based zoning method and features selected using correlation coefficient. The HME classifier consistently outperforms the flat neural network classifier in all category sets. The difference is statistically significant for HD-29, HD-28 and HD-119. Another important feature that we must point out is that our HME model has lower variance in performance in all the category sets. This result confirms the theoretical claim by

[8] categories with no examples in the test set will have $F_1 = 0$ if a document is assigned to the class or $F_1 = 1$ if no documents are assigned to the class.

Table VI. Comparison of categorization performance using the centroid-based and Knn-based category zones. Values are macro-averaged F_1 over the respective set of categories on the test set (50,216 documents).

	centroid-based zone		Knn-based zone	
	Flat NN	HME	Flat NN	HME
HD-49	0.5033	0.5241	0.5042	0.5150
HD-28	0.3589	0.4304	0.3613	0.4159
HD-26	0.5653	0.5794	0.4599	0.4828
HD-119	0.4797	0.5126	0.4542	0.4798

Jordan and Jacobs that soft splitting is a variance reduction method [13]. This is in general a desirable property of a classifier since this indicates performance that is more stable across categories. Comparing the HME against the optimized Rocchio classifier we note that Rocchio significantly outperforms the HME on the HD-49 and HD-28 categories while the HME significantly outperforms Rocchio for the HD-26 categories. However, there is no significant difference between both classifiers on the HD-119 set. As suggested by a reviewer, the good performance of the Rocchio classifier may in part be due to the particular combination of category zoning and feature selection methods used for our classifiers. To remind the reader, first a centroid-based category zone is identified for each category. This zone of documents is then used for feature selection. Although this approach is a "filter" approach (described in section 4) the specifics of the centroid-based category zoning technique suggests that the approach is more of a "wrapper" for the Rocchio classifier while it is definitely a "filter" for the neural net models. Unfortunately the expense of a wrapper approach for the neural net models prohibits us from further exploration of this aspect. This is one of the limitations of the neural net models. Our Rocchio results also emphasize the conclusion by Schapire *et al.* that when the Rocchio classifier is properly trained, it performs as well as other methods [31]. This result is in contrast with the performance of Rocchio classifiers observed by other researchers [16, 19, 44].

A detailed analysis of the behavior of the HME with respect to the flat neural network shows that the threshold for the hierarchical classifier is less than or equal to the threshold for the flat classifier in 95 of the 103 categories. This is an expected result because the intermediate layers perform a pre-filtering of "bad candidate texts" hence the experts

Table VII. Comparison between the flat NN, HME and the optimized Rocchio classifiers.

		Flat NN	HME	Opt-Rocchio
Macro	HD-49	0.5033	0.5241	0.5491
Avg F_1	HD-28	0.3589	0.4304	0.5176
	HD-26	0.5653	0.5794	0.4524
	HD-119	0.4797	0.5126	0.5161
Variance	HD-49	0.02589	0.02298	0.02470
	HD-28	0.06746	0.06773	0.05467
	HD-26	0.17879	0.14583	0.16775
	HD-119	0.08000	0.06754	0.06877

receive a smaller number of examples. Since the optimization process sets these thresholds to maximize the F_1 values in the training set, when the "bad matches" have been filtered the algorithm is able to set a lower threshold that increases the number of true positives without significantly increasing the number of false positives. The idea is to have a hierarchy that is good at filtering false positives.

Table VIII shows the number of documents that pass through each gate in the test set. The number of documents in the test collection is 50,216. The root node filters most of the documents since only $9,238$ pass through it. Observe that gates 1.11, 1.11.8, and 1.11.9 allow a big portion of the documents to pass to the lower levels. This is because they contain two of the categories with the highest number of training examples ("Coronary Diseases", and "Myocardial Infarction"). We expected this to harm the performance of the rest of the categories in these subnodes but in practice this did not happen. For example, only 2 of the 8 categories in the Coronary Diseases subtree (not shown in the table) have a slightly lower performance in the HME model than in the flat classifier.

9.4. Comparing Results with other Published Works

The OHSUMED collection has been used by very few researchers for text categorization. Moreover, to the best of our knowledge only two studies have used its entire set of $14,000$ MeSH categories [15, 42]. The main reason for this is that many text categorization methods do not scale to such a large dataset. Yang [42], Lewis *et al.* [19], and Lam and Ho [16] have published results using the subset of categories from the

Table VIII. Number of documents that pass each gate in the test set.

Level 1		# of doc. ≥ threshold
(root) Heart Diseases		9,238
Level 2	Level 3	
1.1 Arrhythmia		1,464
	1.1.1 Heart Block	176
	1.1.2 Pre-Excitation Syndromes	45
	1.1.3 Tachycardia	583
1.2 Endocarditis		293
	1.2.4 Endocarditis, Baterial	216
1.4 Heart Defects Congenital		1,187
	1.3.5 Heart Septal Defects	277
	1.3.6 Transposition of Great Vessels	57
1.6 Heart Failure,Congestive		1,591
1.7 Heart Rupture		377
1.8 Heart Valve Diseases		1,592
1.9 Myocardial Diseases		4,281
	1.9.7 Cardiomyopathy, Hypertrophic	102
1.10 Pericarditis		447
1.11 Myocardial Ischemia		5,769
	1.11.8 Coronary Diseases	3,343
	1.11.9 Myocardial Infarction	2,570

"Heart Diseases" sub-tree (HD-119). This has become a standard set for comparing results for text categorization in the OHSUMED collection. However, when reading these three works carefully we found that each paper uses a different test set and report results on a different number of categories.

Lewis *et al.* use the set of 183, 229 documents from 1987 to 1990 for training and all the 50, 216 documents from the year 1991 as a test set. Our experiments follow exactly this partition with a further reduction for training (using zoning techniques) but the test set is the same. Table IX shows that our flat neural network model performs at the same level for HD-49 and slightly worse for HD-28, as the Exponentiated Gradient (EG) algorithm in Lewis *et al.* [19]. EG is the second ranked and the top ranked algorithm for HD-49 and HD-28 respectively in [19]. (Note that the last three rows of the table show results from our work that are most

comparable). The HME model on the other hand shows a performance slightly lower (4.7%) than the top ranked Widrow-Hoff (WH) for HD-49 but significantly higher than the best (10.2%) performance for HD-28. Both the flat NN and HME outperform the Rocchio classifier reported as a baseline by Lewis *et al.* [19]. Interestingly, our optimized Rocchio classifier performs at the same level as WH for HD-49 but significantly better than all other classifiers for HD-28.

Yang [42] conducts a very different experiment by reducing the collection to only those documents that are positive examples of the categories of the HD-119. This limits the training set to 12,284 documents and the test set to 3,760 documents. She explains that the reason for such reduction is the scalability of the LLSF method which needs to compute a Singular Values Decomposition, a procedure that cannot be performed efficiently on a large matrix. However, this simplification creates a partition of the OHSUMED collection that is structurally different from the one originally proposed by Lewis *et al.* In order to explore differences we used our trained classifiers on this reduced test set (with the same thresholds as before). The results obtained are 0.525, 0.521 and 0.530 for the flat neural networks, the HME and optimized Rocchio respectively. Observe that for our hierarchical model, Yang's reduction is equivalent to having a perfect classifier at the root node of the tree and thus using only gates at levels 2, 3 and 4. We then ran a second experiment where although we did not retrain the neural networks or Rocchio classifiers on the reduced training set, threshold selection was done only on the set of 12,824 positive examples in the training set (instead of using the whole set of 183,229 documents of the training set) and the gates in levels 2, 3 and 4 were used. The results for our flat neural networks, HME and optimized Rocchio classifiers on this reduced subset are 0.564, 0.557, and 0.558 respectively, scores which are significantly higher than those we found using the Lewis *et al.* partition. These scores are about the same as the ones reported by Yang for the LLSF and ExpNet classifiers, and significantly above Yang's baseline STR classifier. Interestingly the HME model does not outperform the flat neural network model in this constrained experiment. We looked closely at each category and found that this was due to the fact that our originally trained gates discard documents that are relevant to their descendants which then impacts the final performance of the classifier. Although performance may be improved if we train the gates using only the set of positive examples[9], we believe that our original categoriza-

[9] In fact, in experimentation with Knn zoning which uses a smaller training set that more closely resembles the positive examples, HME was better than the flat neural network classifier.

Table IX. Performance comparison between the flat NN, HME and Rocchio classifiers, and classifiers in Yang [42] (Y) and Lewis *et al.* [19] (L).

Method	Yang [42]	Lewis *et al.* [19]	
	HD-119	HD-49	HD-28
Y:LLSF	0.55	–	–
Y:ExpNet	0.54	–	–
Y:STR	0.38	–	–
L:Rocchio	–	0.44	0.33
L:EG	–	0.50	0.39
L:WH	–	0.55	0.39
Flat NN	0.564	0.503	0.358
HME	0.557	0.524	0.430
Opt-Rocchio	0.558	0.549	0.518

tion task is more realistic since we include both positive and negative examples in the test set.

Lam and Ho [16] report results of experiments using the Generalized Instance Set (GIS) algorithm. They use the documents from 1991 and take the first 33,478 documents for training and the last 16,738 documents for testing. In contrast to Yang's reduction, this is more consistent with the partition proposed by Lewis *et al.* because it retains all documents from the collection (not just the positive examples). Lam and Ho report results using micro averaged BEP on the set of 84 categories that have at least one example in both training and test sets. We tested our previously trained HME model in this reduced test set and obtained a micro-averaged BEP value of 0.502 which is significantly lower than their performance of 0.572 for their GIS algorithm with Rocchio generalization. In future research we will train and test our HME model using their data set.

Joachims [10] has also published results for the OHSUMED collection using support vector machines. His work uses the first $20,000$ documents of the year 1991 dividing it into two sets of $10,000$ documents each that are used for training and testing respectively. He reports impressive results but his text categorization task is very different from the ones in the previously discussed works. Joachims assumes that if a category in the UMLS tree is assigned then its more general category in the hierarchy is also present. Although this is similar to

our definition of gates, the difference is that he uses only the high level disease categories. This simplifies the categorization task considerably and probably explains the good results obtained in the reported SVM experiments. The focus on general disease categories alone prevents comparison of Joachims' results with any of the previously published results. We did not run our experiments limited to high level disease categories. However, we consider that the use of SVM can be a good choice for building an architecture like the one we have proposed, and we plan to explore this in our future research.

We believe that the combination of category zoning and feature selection gives a significant boost to Rocchio performance. Optimized versions of the Rocchio classifier in previous work have focused on query zoning and dynamic feedback optimization [31]. We are not aware of any previous work on reducing the set of features used in the centroid vector for text categorization purposes[10]. Observe that our feature selection method favors features indicative of the presence of the category and discards features that indicate the absence of the category. Feature selection has an important impact because similarity computation between the document and the centroid vector are made only on the subspace formed by the selected features.

10. Comparison with Related Work in Hierarchical Categorization

We now focus on methods exploring the hierarchical structures of classification schemes.

Koller and Sahami [14, 30] proposed a hierarchical approach that trains independent Bayesian classifiers for each node of the hierarchy. The classification scheme then starts at the root and greedily selects the best link to a second level classifier. This process is repeated until a leaf is reached or until no child node is a good candidate. Observe that their method selects a single path (the one with highest probability) and assigns all the categories in the path to the document. According to their approach, the hierarchical structure is used as a filter that activates only a single best classification path. Also errors in classification at the higher levels are irrecoverable in the lower levels. They tested their results in the Reuters collection defining as higher nodes in the hierarchy those categories that subsume other categories. Although similar in spirit, our approach differs in the classification assignment model. We separate the identification of general concepts

[10] During the review of this article two works that use feature selection in the context of a Rocchio approach have been published [5, 8]

from the assignment of general categories. Our approach also activates more than a single path in the hierarchy in contrast to their "the winner takes all" approach. There is also an obvious difference in the machine learning algorithm since they use Bayesian classifiers while we use neural networks. However in future work we plan to explore Bayesian classifiers within the HME approach.

Ng *et al.* [27] build their hierarchical classifier using perceptrons. Each node of the hierarchical tree is represented by a perceptron. They distinguish two types of nodes, leaf nodes and non-leaf nodes. They apply this to the Reuters corpus where the categories reflect a geographical/topical hierarchy. Their hierarchy has as a first level all the possible countries, and for each country different topics are defined, *e.g.*, economics and politics. The leaf nodes are specific categories of the second level (*e.g.*, for economics they have communications, industry, *etc.*). The hierarchical classifier receives a document and checks whether it belongs to any of the first level nodes (the root node only connects to the different country nodes). If the tested document activates any of the first level nodes, then the descendant categories of that node are tested recursively. If at any of the non-leaf nodes the process finds that none of its children is a good candidate, then the categorization stops at that branch of the recursion. The output of the classifier is the final set of leaf nodes reached in the recursion (zero, one or more). This is similar to the *Pachinko machine* proposed by Koller and Sahami [14] but with multiple outputs instead of a single output. Our approach is similar to Ng *et al.* [27] in that we also use a top-down approach. The difference is in the type of classifier. We use non-linear classifiers in each node while they use linear classifiers. Although the combination of the linear classifiers in the hierarchy creates a non-linear classifier some studies have shown that this covers a limited number of non linear problems[11] [13, 36]. Our approach also differs significantly in the way feature selection and subset training selection is done. Although their experiments also use correlation coefficient for feature selection their results for the hierarchical classifier are well below other methods that use the Reuters collection. They report F_1 values of 0.52 for the best automatic feature selection method and 0.728 for the manually selected features (which is considerable lower than 0.85 [44]). Their best results with the hierarchical classifier are obtained using manually selected features. We believe that their good results may be a consequence of manual feature selection. Furthermore, our approach based on category zones combined with exploiting the hierarchy is more robust and allows

[11] This is due to the fact that the high level nodes can only create linear boundaries between adjacent expert regions in the input space.

us to get results similar to some of the best methods reported in the literature.

McCallum and his collaborators have been working in text categorization specifically targeting the problem of classifying web pages [22]. Their approach is based on Bayesian classifiers. They use the hierarchical classification structure to improve the accuracy of Bayesian classifiers using a statistical technique called shrinkage that smoothes parameter estimates of a child node with its ancestors in order to obtain more robust estimates. The Bayesian classification schemes involve estimating the parameters of the model from the training collection and then applying the shrinkage method to improve the estimates using the predefined hierarchy of categories. The classification of the test set is performed by computing the posterior probability of each class given the words observed in the test document, and selecting the class with the highest probability. Their experiments show that shrinkage improves the performance when the training data is sparse, reducing the classification error by up to 29%. Our approach is totally different from McCallum's approach in terms of the classification method used, as well as the assumptions in the categorization task. Observe that their approach assumes that a document belongs to a single category and their model reflects it by selecting the most probable classification.

Mladenić [24] also explored hierarchical structures using the Yahoo! hierarchy to classify web pages. Her approach builds a Bayesian classifier. For each node in the subject hierarchy a classifier is induced. To train each of the non-leaf classifiers a set of positive examples is defined as all the positive examples of the node (the intermediate nodes are also valid categories) plus the positive examples of any descendants. These examples are weighted according to their position in the tree. The classification process works as described before for the Bayesian classifiers with the set of categories with predicted probability ≥ 0.95 are assigned. Our approach differs from Mladenić's work in terms of the classifier algorithms used, as well as the way in which the hierarchy is used for feature selection. Her main effort is in creating a weighting scheme for combining probabilities obtained at different nodes of the tree.

The work on topic spotting by Wiener *et al.* [39] inspired us to try our approach using HME. During the review process of this article they published a sequel to their work applied to hierarchical classifiers [38]. They use a meta-topic network using a two level hierarchy for the Reuters collection and present results that are competitive with those of other methods. Our work differs from theirs in the definition of the gates (our gates behave like binary filters while their meta-topic network helps more to weight the contribution of the experts). Our

work also extends the hierarchical model to more than two levels an idea that is suggested but not developed in [38]. Finally, our evaluation on the OHSUMED collection instead of the Reuters collection allows us to test the benefit of exploiting a real hierarchical classification scheme.

We think that our main contribution is a general method for combining the hierarchical structure of a classification with feature selection and category zones. The zones contain small but optimal subsets of documents that yield features suitable for training neural networks. Our approach shows that even with a set of only 25 features per node, we can get results that are comparable with other methods that use larger feature sets[12].

11. Conclusions

This paper presents a machine learning method for text categorization that takes advantage of pre-existing classification hierarchical structures. In response to our first research question we find that exploiting the hierarchical structure via the HME model increases performance significantly. In response to our second research question we find that although the HME approach is equivalent in performance to an optimized Rocchio approach for the full set of categories, the Rocchio approach is better for medium and high frequency categories while HME is better for the remaining low frequency ones. In comparison with previous results with the Rocchio classifier we find that the approach benefits from category zoning for training set selection followed by feature selection. These results confirm the results published by Schapire *et al.* [31] that a carefully trained Rocchio algorithm might perform as well as other more sophisticated methods.

Our method should scale to large test collections and vocabularies because it divides the problem into smaller tasks that can be solved in shorter time. The top-down processing approach tested here was selected specifically with scalability in mind. Category zoning is also very valuable for large collections since it counters the overwhelming presence of negative examples.

With respect to the feature selection methods tested, we found no significant difference in performance between correlation coefficient, odds ratio, and mutual information when they are used for local feature selection. In particular our results with mutual information, which has been reported before as a poor method when used for global feature se-

[12] Most [27, 39, 38] have used feature sets of 200 or more features per category. In [19] the authors have used all features present in the collection.

lection [43], point towards the importance of addressing the weaknesses of feature selection methods on low frequency terms.

To close, the results obtained by the HME model are comparable with those reported previously in the literature, and significantly better than our flat neural network classifier. For future work we would like to try different ways to construct the hierarchical structure such as using support vector machines. We would also like to change the training strategy to use a variation of boosting adapted to the hierarchical approach. We also hope to improve the performance of our classifier by adding richer features (i.e., n-grams, or phrases) that have been reported to help in the text categorization process [24]. Finally, we plan to explore whether the use of different classifiers, such as linear classifiers and SVM show the same improvements obtained using neural networks.

Acknowledgement– We want to thank Professors David Eichmann, Ted Herman, Gregg Oden, and Alberto Segre, from The University of Iowa for their initial comments and feedback that encouraged us to write this article. We also want to thank Dr. Fabrizio Sebastini and the two anonymous reviewers for their excellent comments and suggestion that helped us to significantly improve this article.

References

1. Apte C, Damerau FJ and Weiss SM. Automated learning of decision rules for text categorization. *ACM Transactions on Information systems*, 3(12), pp. 233–251, July 1994.
2. Bridle J. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In F. Fogelman-Soulie & J. Hérault (Eds.), *Neuro-computing: Algorithms, Architectures, and Applications*. New York: Springer-Verlag, 1989.
3. Breiman L, Friedman JH, Olshen RA, and Stone CJ. Classification and regression trees. Belmont, CA: Wadsworth International Group,1984.
4. Buckley C and Salton G. Optimization of relevance feedback weights. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 351–357, Seattle, WA, July 1995.
5. Caropreso MF, Matwin S and Sebastiani F. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In: Amita G. Chin Ed., *Text Databases and Document Management: Theory and Practice*. Idea Group Publishing, 2001, pp. 78-102.
6. Cohen W and Singer Y. Context-sensitive learning methods for text categorization. In *Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 307-315, Zurich, Switzerland, July 1996.

7. d'Alché–Buc F, Zwierski D and Nadal JP. Trio-Learning: a tool for building Hybrid Neural Trees. *International Journal of Neural Systems*, 5(4), pp. 259–274, December 1994.
8. Galavotti L, Sebastiani F and Simi M. Experiments on the use of feature selection and negative evidence in automated text categorization. In: *Proceedings of ECDL-00, 4th European conference on research and Advanced Technology for Digital Libraries*, Lisbon, Portugal, pp. 59-68, 2000.
9. Hersh W, Buckley C, Leone TJ, and Hickman D. OHSUMED: An interactive retrieval evaluation and new large test collection for research. In W. Bruce Croft and C. J. Van Rijsbergen, editors, *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 192–201, Dublin, Ireland, July 1994.
10. Joachims T. *Estimating the generalization performance of a SVM efficiently.* Technical report LS-8 Report 25. Universität Dortmund, Dortmund, December 1999.
11. Joachims T. Text categorization with support vector machines: Learning with many relevant features. Technical Report LS-8 Report 23, University of Dortmund, 1997.
12. John G, Kohavi R and Pfleger K. Irrelevant features and the Subset Selection Problem. In *Machine Learning: Proceedings of the Eleventh International Conference*, pp. 121–129, Morgan Kaufman Publishers: San Francisco, CA, 1994.
13. Jordan MI and Jacobs RA. Hierarchical mixtures of experts and the EM algorithm. Technical report A.I. Memo No. 1440, Massachusetts Institute of Technology, 1993.
14. Koller D and Sahami M. Hierarchically classifying documents using very few words. In *ICML-97: Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 170–178, San Francisco, CA, 1997.
15. Lam W, Ruiz ME and Srinivasan P. Automatic Text categorization and its application to text retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 11:(6), pp. 865–879, 1999.
16. Lam W and Ho CY. Using a generalized Instance Set for Automatic Text Categorization. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval* , pp. 81–89, 1998.
17. Lewis D and Ringuette M. Comparison of two learning algorithms for text categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, 1994.
18. D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 37–50, June 1992.
19. Lewis DD, Schapire RE, Callan JP and Papka R. Training algorithms for linear text classifiers. In *Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 298–303, Zurich, Switzerland, July 1996.
20. Liu H and Motoda H. Less is more. In Huan Liu and Hiroshi Motoda, editors, *Feature Extraction Construction and Selection: A Data Mining Perspective*, chapter 1, pp. 3–12. Kluwer Academic Publishers, Boston, MA, 1998.
21. McCallum A and Nigam K. A comparison of event models for naive Bayes text classification. In *Learning for Text Categorization: Papers from the 1998*

Workshop. AAAI Technical Report WS-98-05, pp. 41–48. AAAI, AAAI Press, July 1998.

22. McCallum A, Rosenfeld R, Mitchell T, and Ng AY. Improving text classification by shrinkage in a hierarchy of classes. In *Proceedings of the 15th International Conference on Machine Learning*. AAAI, Morgan Kaufmann, San Francisco, CA, July 1998.
23. McCullagh P, and Nelder JA. *Generalized Linear Models*. London: Chapman and Hall, 1989.
24. Mladeni ć D. *Machine learning on non-homogeneous, distributed text data*. PhD Dissertation, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia, 1998.
25. Moulinier I and Ganascia JG (1996) Applying an existing machine learning algorithm to text categorization. In: S Wermer, E Riloff and G Scheler Eds., Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing. Springer Verlag, Heidelberg, Germany, pp. 343-354, 1996.
26. National Library of Medicine. *Unified Medical Language System (UMLS) Knowledge Sources*. 10th edition, U.S. Department of Health and Human Services, National Institute of Health, National Library of Medicine, January 1999.
27. Ng HT, Goh WB, and Low KL. Feature selection, perceptron learning, and a usability case study for text categorization. In Nicholas Belkin, A. Desai Narasimhalu, and Peter Willett, editors, *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Philadelphia, PA, pp. 67–73, July 1997.
28. RocchioJJ. Relevance Feedback in Information Retrieval. In Gerald Salton Ed., *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice Hall, Englewood Cliffs, New Jersey, 1971.
29. Rumelhart DE, Durbin R, Golden R, and Chauvin Y. Backpropagation: The basic Theory. In P. Smolensky, M. C. Mozer and D. E. Rumelhart (eds), *Mathematical Perspectives on Neural Networks*, Lawrence Earlbaum Associates, Hillsdale, NJ, pp. 533–566, 1996.
30. Sahami M. *Using Machine Learning to Improve Information Access*. PhD thesis, Stanford University, Computer Science Department, 1998.
31. Schapire RE, Singer Y, and Singhal A. Boosting and Rocchio applied to text filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia, pp. 215–223, August 1998.
32. Schütze H, Hull DA and Pedersen JO. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, WA, pp. 229–237, July 1995.
33. Singhal A, Buckley C and Mitra M. Pivoted document length normalization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Zurich, Switzerland, pp. 21–29, August 1996.
34. Singhal A, Mitra M, and Buckley C. Learning routing queries in a query zone. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Philadelphia, PA, pp. 25–32, July 1997.
35. van Rijsbergen CJ. *Information Retrieval*. Butterworths, London, 2nd edition, 1979.

36. Waterhouse SR. Classification and regression using mixtures of experts. PhD. Dissertation, University of Cambridge, Cambridge England, 1997.
37. Waterhouse SR and Robinson AJ. Classification using Hierarchical Mixtures of Experts. In *Proceedings of the 1994 IEEE Workshop on Neural Networks for Signal Processing IV*, pp. 177–186, 1994.
38. Weigend AS, Wiener ED and Pedersen JO. Exploiting Hierarchy in Text Categorization. *Information Retrieval*, 1(3), pp. 193-216, 1999.
39. Wiener E, Pedersen JO, and Weigend AS. A neural network approach to topic spotting. In *Proceedings of SDAIR'95*, pp. 317–332, 1995.
40. Wolpert DH. Stacked generalization, Tech. Rep. LA-UR-90-3460, The Santa Fe Institute, Santa Fe, NM, 1993.
41. Yang J and Honovar V. Feature subset selection using a genetic algorithm. In Huan Liu and Hiroshi Motoda, editors, *Feature Extraction, Construction and Selection: A Data Mining Perspective*, chapter 8, pp 117–136. Kluwer Academic Publishers, Boston, MA, 1998.
42. Yang Y. An evaluation of statistical approaches to MEDLINE indexing. In *Proceedings of the American Medical Informatic Association (AMIA)*, pp. 358–362, 1996.
43. Yang Y and Pedersen JO. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*. Morgan Kaufmann Publishers, San Francisco, CA, July 1997.
44. Yang Y. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1), pp. 69–90, 1999.