Performance Measurement Framework for Hierarchical Text Classification ⋆

Aixin Sun, Ee-Peng Lim, Wee-Keong Ng

Centre for Advanced Information Systems, Nanyang Technological University, Nanyang Avenue, Singapore 639798
*Email:*sunaixin@pmail.ntu.edu.sg aseplim@ntu.edu.sg

Abstract Hierarchical text classification or simply hierarchical classification refers to assigning a document to one or more suitable categories from a hierarchical category space. In our literature survey, we have found that the existing hierarchical classification experiments used a variety of measures to evaluate performance. These performance measures often assume independence between categories and do not consider documents misclassified into categories that are similar or not far from the correct categories in the category tree. In this paper, we therefore propose new performance measures for hierarchical classification. The proposed performance measures consist of *category similarity measures* and *distance based measures* that consider the contributions of misclassified documents. Our experiments on hierarchical classification methods based on SVM classifiers and binary Naïve Bayes classifiers showed that SVM classifiers perform better than Naïve Bayes classifiers on Reuters-21578 collection according to the extended measures. A new classifier-centric measure called *blocking measure* is also defined to examine the performance of subtree classifiers in a top-down level-based hierarchical classification method.
Keywords Hierarchical Text Classification, Text Categorization, Text Mining, Performance Measure, Performance Evaluation.

1 Introduction

1.1 Motivation

Organizations today conduct many of their activities electronically and generate enormous amount of electronic documents as part of their internal and external activities. To a user, it is extremely difficult to locate information within these large collections of documents without some form of text classification. Text Classification(TC) or Text Categorization is the process of automatically assigning one or more predefined categories to text documents. In the text classification research, most of the studies have focused on *flat classification* where the predefined categories are treated in isolation and there is no structure defining the relationships among them (Yang 1999). Such categories are also known as *flat categories*. However, when the number of categories grows to a significantly large number, or when the number of documents in each category reaches a large number, the information overloading problem occurs once again but now at the category level. For example, when the thousands or even tens of thousands of categories in the Yahoo![1] hierarchy are mapped into a flat space, it will require considerable amount of time to locate the categories for browsing and searching. By organizing the categories in a hierarchy, Yahoo! has made it easier to look for information.

In most of these methods, the categories are organized in tree-like structures. On the whole, we can identify four distinct category structures for hierarchical classification. They are:

1. *Virtual category tree:* In this category structure, categories are organized as a tree. Each category can belong to at most one parent category and documents can only be assigned to leaf categories only. Most of the existing hierarchical classification methods have adopted virtual category trees as their category structure (Greiner, Grove & Schuurmans 1997, Dumais & Chen 2000, Sasaki & Kita 1998). Dumais and Chen showed that the performance of hierarchical classification for a virtual category tree is better than that of classification for flat categories corresponding to only leaf categories of the virtual category tree (Dumais & Chen 2000).
2. *Category tree:* This is an extension of the virtual category tree that allows documents to be assigned into both internal categories and leaf categories (Wang, Zhou & Liew 1999, Wang, Zhou & He 2001).
3. *Virtual directed acyclic category graph:* In this category structure, categories are organized as a Directed Acyclic Graph (DAG). Similar to the virtual category tree, documents can only be assigned to leaf categories.
4. *Directed acyclic category graph:* This is perhaps the most commonly-used structure in the popular web

⋆ The work is partially supported by the SingAREN21 research grant M48020004, Singapore.

[1] http://www.yahoo.com

directory services such as Yahoo! and Open Directory Project[2]. Documents can be assigned to both internal and leaf categories (Labrou & Finin 1999).

In the four kinds of category structures, categories that do not hold documents are defined as *virtual categories* while the ones that hold documents (e.g., the leaf categories in virtual category tree) are known as *real categories*.

In addition to category structures, further *constraints* can be imposed on hierarchical classification to ensure that the classified documents satisfy some common properties. By considering different constraints, the hierarchical classification methods will give quite different results. For example:

- When parent-child relationship between categories represents a strict subsumption relationship, a document cannot be assigned to both a child category and its parent (ancestor) simultaneously. In this case, the document should be assigned to only the child category.
- When parent-child relationships between a category and all its children represent a total subsumption relationship, a document should be assigned to the parent category if it is assigned to all the child categories.

There are basically two approaches adopted by the existing hierarchical classification methods: **big-bang approach** and **top-down level-based approach**. In the big-bang approach, a document is classified (or rejected) into (or from) a category in the category tree by a classifier in *one* single step (Labrou & Finin 1999, Sasaki & Kita 1998, McCallum, Rosenfeld, Mitchell & Ng 1998, Wang et al. 2001, Gaussier, Goutte, Popat & Chen 2002, Vinokourov & Girolami 2002). The same classifier may be used to assign a document to more than one category in the category tree but each document can only be handled by one classifier. The assigned categories can be internal or leaf categories depending on the category structure supported by the method.

In the top-down level-based approach, one or more classifiers are constructed at each level of the category tree and each classifier works as a flat classifier at that level. A document will first be classified by the classifier at the root level into one or more lower level categories. It will then be further classified by the classifier(s) at the lower level category(ies) until it reaches one or more final categories which could be leaf categories or internal categories. A few hierarchical classification methods have been proposed recently (D'Alessio, Murray, Schiaffino & Kershenbaum 2000, Dumais & Chen 2000, Mladenic 1998, Sasaki & Kita 1998, Wang et al. 2001, Weigend, Wiener & Pedersen 1999). These methods will be described in more detail in Section 2.

As more and more hierarchical classification methods have been or will be developed, it is becoming important to establish a common framework to evaluate the performance of these different classification methods. In the context of flat classification, there are well accepted performance measures defined to evaluate and compare flat classification methods. There are also publicly available text collections for conducting flat classification experiments (Lewis 1995, Robertson & Hull 2000). Similar framework, however, does not really exist for hierarchical text classification.

In existing hierarchical classification research, most of the experiments have been conducted in various different ways using different performance measures and text collections (D'Alessio et al. 2000, Dumais & Chen 2000, Mladenic 1998, Sasaki & Kita 1998, Wang et al. 2001, Weigend et al. 1999). Many directly used the performance measures for flat classification to evaluate the performance of hierarchical classification methods, ignoring the relationships between categories in the category tree.

In this paper, we define a performance measurement framework for hierarchical classification. In flat classification, precision and recall have been widely used as the performance measures. To capture the performance of hierarchical classification methods accurately, more complicated versions of the precision and recall should be developed to cope with the unique relationships between the categories in a category tree. In general, the categories from the same subtree share more domain knowledge than the ones from different subtrees, that is, the categories from the same subtree are semantically closer to one another. For example in (Larkey & Croft 1996), *partial success* is defined when a document is correctly assigned to a main category but not the sub-categories. The partial success definition clearly makes use of parent-child relationship of the hierarchy. By not considering the "closeness" of categories, the performance of hierarchical classification may not be accurately captured.

In this research pursuit, the performance measures for hierarchical classification methods should therefore satisfy the following criteria.

- The performance measures should be natural extensions of those used in flat classification. This will give some consistency to their meaning and use.
- The new performance measures should use the information carried by the hierarchy, such as the relationships between categories, as such relationships affect the effectiveness of hierarchical classification from the user perspective.

For top-down level-based hierarchical classification methods, more than one classifier will be required, and the performance of a classifier may affect the others. The final decision of which category(ies) should be assigned to a document is made by a series of classifiers arranged in top-down manner at different levels. Any non-performing classifier in this series will lead to poor

[2] http://dmoz.org

performance of the classifiers at the lower levels. Such a non-performing classifier is therefore a performance bottleneck. As part of our research, a special kind of classifier-centric measure is therefore required to identify the non-performing classifiers in a hierarchical classification method.

1.2 Objectives and Contribution

In this paper, the performance measurement issues of hierarchical text classification are studied. Our objectives and contributions are summarized as follows:

1. *Performance measures* for hierarchical classification. We propose new performance measures based on relationships among categories in a hierarchy. The *similarity based measures* are defined based on the semantic relationships while the *distance-based measures* are defined based on the parent-child relationships. Both types of performance measures originate from the performance measures for flat classification, e.g., the standard precision and recall. The extended measures have been successfully used to capture the performance of hierarchical classification methods in our experiments.
2. Development of new hierarchical classification methods based on Support Vector Machines (SVM) and Naïve Bayes. SVM and Naïve Bayes represent the new and baseline classification methods respectively for text classification. By using these basic classifiers for top-down level-based hierarchical classification methods, we would like to compare their performance using the proposed new measures.
3. *Blocking measure* in top-down level-based hierarchical classification. This classifier-centric measure can be used to quantify the performance of high-level classifiers in top-down level-based hierarchical classification methods.

Our research is built upon the observation that the standard measures used in flat classification do not truly reflect the performance of a hierarchical classification method when a category can be more closely related to some categories in the category hierarchy, while more distantly related to other categories in the hierarchy. For example, banking news category is quite closely related to investment news category, but is distantly related to sport news. These category differences can often be illustrated by how far apart the categories are in a category hierarchy. Furthermore, a banking news document misclassified under investment category should be treated more favorably than the same document misclassified under the sport category. Should a misclassified document appear in an incorrect category nearby the correct one, it will be easier for the user to locate it by browsing the nearby incorrect category. In some cases, a user may have already planned to view categories nearby the correct category. In other cases, the browser or search engine software can be intelligent enough to recommend nearby categories to the user. Hence, a misclassified document in the nearby incorrect category will have a higher chance to be found.

1.3 Outline of Paper

The rest of the paper is organized as follows. We will first give an overview of the related hierarchical classification work in Section 2. In Section 3, the commonly used performance measures in flat classification will be discussed. We will also present our proposed performance measures for hierarchical classification. Our proposed blocking measure will be given in Section 4. In Section 5, we will describe a hierarchical classification method using the SVM and binary Naïve Bayes classifiers. Finally, we conclude our work in Section 6.

2 Related Work

There has not been much work done on a common performance measurement framework for hierarchical classification. One of the reasons could be that hierarchical classification problem is relatively new. Only a few hierarchical classification methods have been proposed so far. Most experiments had been conducted using different data sets under different assumptions about the category structure. We therefore survey these different hierarchical classification methods and their performance measurement experiments.

2.1 Big-Bang Approach

In the big-bang approach, a document can be classified to any categories in the category tree using a single classification step.

Labrou and Finin, in (Labrou & Finin 1999), developed a Rocchio-like classifier to classify documents into a category DAG. For each category, a weighted vector (also known as the category profile) is derived from the category name, the titles and short descriptions of the training documents under the category. The similarity between the test documents and categories will then be computed as the cosine of the document vectors and the weighted category vectors. Using this classification method, each document is assigned to only one category.

Sasaki and Kita, on the other hand, used a rule learning algorithm known as RIPPER (for Repeated Incremental Pruning to Produce Error Reduction) proposed in (Cohen 1995) to perform hierarchical classification (Sasaki & Kita 1998). The category structure used is a virtual category tree. In this method, a set of rules is first generated based on the training documents. Each rule indicates, for a category, the word(s) that has to

appear in a document before the document can be assigned to it. This classification method, again, assigns only one category to each document.

In both (Labrou & Finin 1999) and (Sasaki & Kita 1998), the performance measures in their experiments have been very much based on simple empirical observations of the number of correctly classified documents or the percentage of wrongly classified documents. These measures are not general enough to determine the performance of different hierarchical classification methods. They are also incompatible with the standard measures for flat classification.

A novel hierarchical classification method built upon association rule mining was proposed by Wang in (Wang et al. 2001, Wang et al. 1999). Similar to the method by Sasaki and Kita, this method is able to generate rules to classify documents into a category tree (not virtual tree) by examining the features in the documents. It constructs classification rules of the form $\{t_{i_1}, \ldots, t_{i_n}\} \rightarrow \{C_{i_1}, \ldots, C_{i_q}\}$ from training documents, and applies these rules to test documents. In the above rule format, t_{i_j} and C_{i_k} represent indexed term and category respectively. Wang's hierarchical classification method allows multiple categories to be assigned to a document. Experiments have also been conducted to determine the performance of the proposed method. The performance measure used is *classification error* that represents the percentage of documents that have been incorrectly classified.

Toutanova et al. extended the standard Naïve Bayes classifier and proposed a hierarchical mixture model in (Toutanova, Chen, Popat & Hofmann 2001). The hierarchy of topics is used to provide estimates for class-conditional term probabilities and to obtain a differentiation of words in the hierarchy according to their level of generality/specificity. The inner nodes of the hierarchy represent abstraction levels with their corresponding specific vocabulary. Each word in a document is assumed to be generated from abstraction level on the path from the document class node to the root. They evaluated their model on two virtual trees constructed from 20 Newsgroups and Reuters-21578 respectively. Experimental results reported are micro-averaged precision, recall and F_1 measures. Compared to the Hierarchical Shrinkage model proposed by (McCallum et al. 1998) that is similar to their model, the hierarchical mixture model achieved better results on Newsgroup when limited number of training documents are given.

2.2 Top-Down Level-Based Classification

In the top-down level-based classification, the classification is accomplished with the cooperation of all classifiers built at each level of the category tree. There is one or more classifiers at each level. The test document is first classified at the root category of the hierarchy. It is then classified to categories at the next level. This process will continue on until the document reaches leaf category(ies) or internal category(ies) at which the document cannot be further classified.

In (D'Alessio et al. 2000), D'Alessio et al. described categories by features derived from the training documents using an algorithm known as ACTION (for Automatic Classification for Full-Text Documents). At each category, a binary or one-of-M (m-ary) classifier is used to determine whether a document should belong to the category or one of its children. Three performance measures, i.e., precision, recall and F-measure have been used in their experiments. Note that these are among the standard ones for flat classification.

In the work by Koller and Sahami, three category trees are extracted from the Reuters collection for some selected category labels (Koller & Sahami 1997). These category trees are of the height of 2. A hierarchical classification method using multiple Bayesian classifiers was applied on these three category trees. Each test document is passed to the first level classifiers before the ones at the second level. The document will be assigned to the child category only if both the parent and child classifiers accept it. It was shown that this method outperforms that for flat classification when there is only a small number of features selected for each category. If large number of features are selected, the performance of the hierarchical classification method is comparable to that of the flat one.

Dumais and Chen proposed the use of Support Vector Machines (SVM) classifiers to classify web pages into a virtual category tree using top-down level-based approach (Dumais & Chen 2000). Their method allows a web page to be assigned to a child category even if the former is not favored by the parent category. The category structure used is a virtual category tree with depth 2 obtained from the LookSmart's web directory[3]. As the hierarchy is a virtual category tree, only leaf categories can hold documents. In their experiments, the SVM classifiers are employed in both the hierarchical classification and flat classification. It was found that the performance of hierarchical classification method enjoyed a better accuracy. The performance measure used in the experiments was F-measure. The problem of blocking was mentioned in the paper, but there was no measures defined to evaluate the extent of blocking.

2.3 Discussions

Unlike the top-down level-based approach, the big-bang approach uses information carried by the category structure during the training phase but not the classification phase. Classifiers in the big-bang approach assign the most suitable categories to the test documents irrespective of their locations in the category structure. With

[3] http://www.looksmart.com

Table 1 Contingency table for category C_i

Category C_i		Expert Judgments YES	Expert Judgments NO
Classifier	YES	TP_i	FP_i
Judgments	NO	FN_i	TN_i

top-down level-based approach, each document is first considered at the root of the category tree before the low-level categories. During this process, *subtree classifiers* are employed to determine whether the document belongs to the corresponding subtrees. The other classifiers, known as *local classifiers*, will determine the final assignment of the document to the categories. Various type of classifiers (e.g., binary or m-ary) can be used in the classification and these classifiers can learn their own discriminative features independently or cooperatively. One of its obvious problems is that a misclassification at a parent (ancestor) category may force a document to be discarded before it can be classified into the child categories. In other words, some recovery mechanism may be required to remedy the error made by the parent (ancestor) classifier. This is known as the "blocking" problem in this paper. Classification methods based on top-down approach also require more training examples since multiple classifiers have to be constructed each requiring a different training set. Without adequate training examples, the performance of these classifiers may suffer.

3 Performance Measures

3.1 Performance Measures for Flat Classification

In the text classification survey by Sebastiani (Sebastiani 2002), the most commonly used performance measures in flat classification are the classic Information Retrieval (IR) notions of *Precision* and *Recall*. Precision for a category C_i, denoted by Pr_i, measures the percentage of correct assignments among all the documents assigned to C_i. The Recall Re_i for C_i gives the percentage of correct assignments in C_i among all the documents that should be assigned to C_i. Pr_i and Re_i are also known as the standard precision and recall respectively for C_i in this paper. The contingency table for category C_i from the category space $\{C_1, \ldots, C_m\}$ is shown in Table 1. Let TP_i be the set of documents correctly classified into category C_i; FP_i be the set of documents wrongly accepted; FN_i be the set of documents wrongly rejected and TN_i be the set of documents correctly rejected. The standard precision and recall are defined below[4]:

$$Pr_i = \frac{|TP_i|}{|TP_i| + |FP_i|} \tag{1}$$

$$Re_i = \frac{|TP_i|}{|TP_i| + |FN_i|} \tag{2}$$

[4] In all the formulas presented in this paper, $|S|$ gives the number of elements in set S.

Based on the standard precision and recall for each category, the overall precision and recall for the whole category space, i.e., $\{C_1, \ldots, C_m\}$, can be obtained in two ways: *micro-average* and *macro-average*. Micro-average gives equal importance to each document, while macro-average gives equal importance to each category (Yang 1999).

1. *Micro-Average*: The overall precision and recall of the category space $\{C_1, \ldots, C_m\}$ are obtained from the overall number of documents correctly accepted, wrongly accepted, and wrongly rejected.

$$\hat{Pr}^{\mu} = \frac{\sum_{i=1}^{m} |TP_i|}{\sum_{i=1}^{m} (|TP_i| + |FP_i|)} \tag{3}$$

$$\hat{Re}^{\mu} = \frac{\sum_{i=1}^{m} |TP_i|}{\sum_{i=1}^{m} (|TP_i| + |FN_i|)} \tag{4}$$

2. *Macro-Average*: The overall precision and recall of the category space $\{C_1, \ldots, C_m\}$ are computed by averaging the precision and recall for all the categories in the category space.

$$\hat{Pr}^{M} = \frac{\sum_{i=1}^{m} Pr_i}{m} \tag{5}$$

$$\hat{Re}^{M} = \frac{\sum_{i=1}^{m} Re_i}{m} \tag{6}$$

Neither precision nor recall is useful as a performance measure in isolation (Sebastiani 2002). In fact, the precision value will increase as the condition of assigning a category to a document becomes more stringent. On the other hand, the recall value will decrease. Therefore, the performance of the text classification has often been measured by the combination of the two measures. The popular combinations are listed below:

1. *Break-Even Point*(BEP): BEP, proposed by Lewis (Lewis 1992), defines the point at which precision and recall are equal. The BEP is usually obtained by interpolating the precision and recall values in the precision-recall graph.
2. F_β *Measure*: F_β measure was proposed by Rijsbergen (Rijsbergen 1979). This elegant measure computes a single score from precision and recall values according to the user-defined importance (i.e., β) of precision and recall. The formula is:

$$F_\beta = \frac{(\beta^2 + 1) \cdot P_r \cdot R_e}{\beta^2 \cdot P_r + R_e} \quad \text{where } \beta \in [0, \infty) \tag{7}$$

When $\beta = 0$, F_0 is the same as precision. If β approaches ∞, F_∞ is the same as recall. The F_β is the same as BEP if P_r and R_e are equal. Normally, $\beta = 1$ is used (Yang 1999), that is, the precision and recall are of the same importance.

3. *Average 11-Point Precision*: The precision values are interpolated at 11 points at which the recall values are 0.0, 0.1, ..., 1.0. This measure is mostly used in the situation when the classification method ranks documents according to their appropriateness to a category or ranks categories by their appropriateness to a document (Sebastiani 2002).

Besides precision and recall, the other commonly-used performance measures include *accuracy* and *error* (Sebastiani 2002, Koller & Sahami 1997), denoted by Ac_i and Er_i respectively. Both accuracy and error can be computed from the contingency table given in Table 1.

$$Ac_i = \frac{|TP_i| + |TN_i|}{|TP_i| + |TN_i| + |FP_i| + |FN_i|} \quad (8)$$

$$Er_i = \frac{|FP_i| + |FN_i|}{|TP_i| + |TN_i| + |FP_i| + |FN_i|} = 1 - Ac_i \quad (9)$$

3.2 Performance Measures for Hierarchical Classification

In this section, we propose new performance measures that consider the inter-category relationships in a category tree. These performance measures are derived from similarity between categories and distance between categories in the hierarchy respectively.

3.2.1 Performance Measures Based on Category Similarity Performance measures based on category similarity are derived based on documents that have been correctly classified and the ones that misclassified. Intuitively, if a classification method A misclassifies documents into categories that are similar to the correct ones, it is considered to be better than another method, say B, that misclassifies the documents into totally unrelated categories. With the standard precision and recall measures, the precision and recall values for A may not be better than B. In the following, we therefore extend the standard precision and recall definitions to distinguish the performance of A and B.

The *category similarity* between two categories C_i and C_k, denoted by $CS(C_i, C_k)$, reflects how "close" the two categories are in terms of semantic. Category similarities in a hierarchical structure can be either manually assigned or derived from the features of the categories, for example, cosine similarity between the two categories. The pairwise category similarity should be provided together with the hierarchy to help user browse information easily. Knowing the pairwise category similarities, one can define the *Average Category Similarity*(ACS).

$$ACS = \frac{2 \times \sum_{i=1}^{m} \sum_{k=i+1}^{m} CS(C_i, C_k)}{m \times (m-1)} \quad (10)$$

Based on the category similarity, we can now measure the degree of correctness of the classification result. Let d_j be a test document, $d_j.agd$ be the assigned categories to d_j, and $d_j.lbd$ be the labelled (correct) categories for d_j. In the simplest case, if d_j is assigned to C_i and the assignment is correct, i.e., $d_j \in TP_i$, d_j will contribute 1 to $|TP_i|$ in the computation of precision or recall for C_i. However, if d_j is wrongly assigned to C_i (i.e., $d_j \in FP_i$), we should consider whether the d_j's labelled categories are similar to C_i. Based on this similarity, we derive the amount of contribution from d_j to C_i. Similarly, if d_j is wrongly rejected from C_i, the *contribution* from d_j to C_i will depends the category similarities between C_i and the assigned categories of d_j. Formally, the *contribution* of document d_j to a category C_i, denoted by $Con(d_j, C_i)$, is defined when d_j is misclassified (e.g., $d_j \in FN_i$ or $d_j \in FP_i$). With respect to the contingency table shown in Table 1, $Con(d_j, C_i)$ is defined as follows:

$$Con(d_j, C_i) = \begin{cases} \min(1, \max(-1, \frac{\sum_{C' \in d_j.agd}(CS(C', C_i) - ACS)}{1 - ACS})) \\ \text{where } d_j \in FN_i \\ \min(1, \max(-1, \frac{\sum_{C' \in d_j.lbd}(CS(C', C_i) - ACS)}{1 - ACS})) \\ \text{where } d_j \in FP_i \end{cases} \quad (11)$$

Note that a document can belong to or be assigned to more than one category. For the documents that are wrongly rejected, the more the document is assigned to a category similar to the correct one, the higher is its contribution towards the correct category. For the documents that are wrongly assigned to a category, the more the labelled category of the document is similar to the assigned one, the higher is the document's contribution towards the assigned category. The contribution of a document can be positive or negative depending on how similar its correct and assigned categories are in comparison with the average category similarity ACS. To prevent one document from being over-rewarded or over-punished, the *contribution* of each document to a category is restricted in the range of $[-1, 1]$ with the $max()$ and $min()$ functions.

Considering all documents in FP_i, the total contribution by documents incorrectly classified under C_i, $FpCon_i$, is defined by:

$$FpCon_i = \sum_{d_j \in FP_i} Con(d_j, C_i)$$

and similarly, the total contribution by documents incorrectly rejected from C_i, $FnCon_i$, is defined by:

$$FnCon_i = \sum_{d_j \in FN_i} Con(d_j, C_i)$$

The extended *precision* and *recall* for category C_i based on category similarity are defined as follows:

$$Pr_i^{CS} = \frac{\max(0, |TP_i| + FpCon_i + FnCon_i)}{|TP_i| + |FP_i| + FnCon_i} \quad (12)$$

$$Re_i^{CS} = \frac{\max(0, |TP_i| + FpCon_i + FnCon_i)}{|TP_i| + |FN_i| + FpCon_i} \quad (13)$$

Since both $FpCon_i$ and $FnCon_i$ can be negative, $|TP_i| + FpCon_i + FnCon_i$ can be negative. Therefore, a max function is applied to the numerator to make it not

less than 0. As $FpCon_i \leq |FP_i|$, when $|TP_i| + |FP_i| + FnCon_i \leq 0$, the numerator $max(0, |TP_i| + FpCon_i + FnCon_i) = 0$ and Pr_i^{CS} can be treated as 0 in this case. The same constraint also applies to Re_i^{CS}.

Compared to the standard precision and recall definitions, the extended precision and recall account for additional contributions by misclassified documents. They nevertheless still stay within the range between 0 and 1. If the extended precision and recall definitions are applied to the flat classification where the category similarities are not considered, they will behave the same as the standard precision and recall. In that case, the category similarity between any two categories, say C_i and C_k is the same. That is, $CS(C_i, C_k) = ACS$. The contributions from the misclassified documents will be 0 according to the contribution definition, and hence $FpCon_i = 0$ and $FnCon_i = 0$.

The *Micro-Average* and *Macro-Average* can be extended based on category similarity.

Micro-Average:

$$\hat{Pr}^{\mu_{CS}} = \frac{\sum_{i=1}^{m}(\max(0, |TP_i| + FpCon_i + FnCon_i))}{\sum_{i=1}^{m}(|TP_i| + |FP_i| + FnCon_i)} \quad (14)$$

$$\hat{Re}^{\mu_{CS}} = \frac{\sum_{i=1}^{m}(\max(0, |TP_i| + FpCon_i + FnCon_i))}{\sum_{i=1}^{m}(|TP_i| + |FN_i| + FpCon_i)} \quad (15)$$

Macro-Average:

$$\hat{Pr}^{M_{CS}} = \frac{\sum_{i=1}^{m} Pr_i^{CS}}{m} \quad (16)$$

$$\hat{Re}^{M_{CS}} = \frac{\sum_{i=1}^{m} Re_r^{CS}}{m} \quad (17)$$

Similar to the extended precision and recall, the extended *accuracy* and *error* for category C_i can be defined based on document contribution.

$$Ac_i^{CS} = \frac{|TP_i| + |TN_i| + FpCon_i + FnCon_i}{|TP_i| + |TN_i| + |FP_i| + |FN_i|} \quad (18)$$

$$Er_i^{CS} = \frac{|FP_i| + |FN_i| - FpCon_i - FnCon_i}{|TP_i| + |FN_i| + |FP_i| + |FN_i|} \quad (19)$$

Note that the sum of extended accuracy and error is 1 which is the same as the original definitions.

3.2.2 Performance Measures based on Distance Instead of using category similarity, we can define performance measures based on the distances between categories in a category tree. The distance between two categories C_i and C_k, denoted by $Dis(C_i, C_k)$, is the number of links between C_i and C_k. Intuitively, the shorter the distance, the closer the two categories.

The distance between categories was first proposed to measure misclassification in (Wang et al. 1999). Nevertheless, the work did not define performance measures based on category distance. To define the *contribution* of misclassified documents, an *acceptable distance*, denoted by Dis_θ, must first be specified by the user. Dis_θ must be greater than 0. For example, if $Dis_\theta = 1$, a misclassification of document that involves the correct and assigned categories at more than 1 link apart will yield negative contribution, but no contribution at 1 link apart. Formally, the contribution of a document d_j to category C_i based on category distance is defined as follows:

$$Con(d_j, C_i) = \begin{cases} \min(1, \max(-1, \sum_{C' \in d_j.agd}(1.0 - \frac{Dis(C', C_i)}{Dis_\theta}))) \\ \text{where } d_j \in FN_i \\ \min(1, \max(-1, \sum_{C' \in d_j.lbd}(1.0 - \frac{Dis(C', C_i)}{Dis_\theta}))) \\ \text{where } d_j \in FP_i \end{cases} \quad (20)$$

In this *contribution* definition, the distance between the labelled and the assigned categories $Dis(C', C_i)$ is compared to the acceptable distance Dis_θ. For instance, a document d_j is wrongly assigned to (and only assigned to) C' while the labelled category is C_i, i.e., $d_j \in FN_i$. The contribution of d_j to category C_i, $Con(d_j, C_i)$, can be in one of the following 3 possible ranges.

1. $0 < Con(d_j, C_i) < 1$ where $0 < Dis(C', C_i) < Dis_\theta$;
2. $Con(d_j, C_i) = 0$ where $Dis(C', C_i) = Dis_\theta$;
3. $-1 \leq Con(d_j, C_i) < 0$ where $Dis(C', C_i) > Dis_\theta$.

In case 1, the document d_j is assigned to category C' that is not too far from the labelled category C_i and the contribution is positive. When $Dis(C', C_i) = Dis_\theta$ occurs, the assignment of the d_j is still acceptable, and should not be punished, hence $1.0 - \frac{Dis(C', C_i)}{Dis_\theta} = 0$. Once d_j is assigned to a farther category in case 3, the contribution of the d_j is negative. With this new definition for *contribution*, the extended precision and recall based on category distance, denoted by Pr_i^{DB} and Re_i^{DB}, can be defined using the formulas (12) and (13) respectively. With the same reason discussed in Section 3.2.1, both Pr_i^{DB} and Re_i^{DB} will stay in the range between 0 and 1. Similarly, *Micro-Average*, *Macro-Average*, F_β, *accuracy* and *error* can be extended accordingly.

It should be pointed out that Dis_θ is a parameter for performance measure. A larger Dis_θ will give higher precision and recall compared to a smaller Dis_θ. Two criteria can be adopted for choosing a proper Dis_θ; one is the size (depth) of hierarchy and the other is the nature of the document to be classified. The first criterion is obvious and normally a large Dis_θ is chosen for a large hierarchy. For example, if the document set to be classified contains news articles, we can choose a large Dis_θ. However if the data are medical records, a relatively small Dis_θ should be used. The reason is that the misclassifications of medical records are more harmful than that of news articles. The value of Dis_θ should be specified when the precision and recall based on category distance are presented.

3.2.3 Discussions As mentioned in Section 3.1, it is not sufficient to evaluate a classification method using either

precision or recall only. Instead, they have to be considered together. The performance measures that combine both precision and recall are the Break-Even Point (BEP), F_β and Average 11-Point Precision. Among them, F_β can be easily computed using the extended precision and recall. BEP can be applied to classification methods that can rank documents for each category. In hierarchical classification using the big-bang approach, BEP and Average 11-Point Precision can be computed for classification methods that can rank *all* documents in the test set for each category. On the other hand, for those classification methods using top-down level-based approach, multiple classifiers may be involved in the classification method, the test documents available for classification at a level are determined by the parent classifier as the latter may reject documents before they reach the child classifiers. With such restriction, it is difficult to compute the BEP and Average 11-Point Precision for each category. Hence, we argue that the above two performance measures are less applicable to the hierarchical classification methods based on top-down approach.

Wang pointed out that misclassification errors in hierarchical classification may not be symmetric (Wang et al. 2001). That is, the classification errors made at the high-level categories are more serious than the ones made at the low level categories in the hierarchy. This is true if the hierarchy is designed such that the higher-level categories contain information strictly more general than that in the lower-level categories. The above requirement however may not be observed by all the category trees. We therefore treat all the errors symmetrically. On the other hand, the *asymmetric measures* can be developed by extending the symmetric ones. For example, the measures based on category distance can be easily extended by defining different values for travelling links upward from a child category to a parent category(e.g., $distance_{up} = 1.5$) and downward from parent to child (e.g., $distance_{down} = 1$). The details on extending the symmetric measures is beyond the scope of this paper.

4 Blocking Measure in Top-Down Level-Based Approach

In top-down level-based hierarchical classification, the documents are classified by the classifiers at high levels followed by the ones at low levels. Although the classifiers at the root level and internal levels may not classify documents into the final categories, they do play important roles in facilitating the correct classification to happen at the final categories. The errors made by the high level classifiers are not recoverable by the low level classifiers unless some special error recovery mechanism is incorporated into the method. For example, if a document is wrongly rejected by the classifier at root level, it will not have chance to be further classified by any other classifiers and hence not be assigned to any categories in the tree. Therefore, for each category that holds documents, there are two reasons for some of the documents not assigned to it by the classification method in top-down level-based hierarchical classification. One is that the documents are rejected by the local classifier for the category and the other is that the documents are rejected by a higher level classifier. In this paper, we define the blocking factor as a means to measure the extent of documents wrongly rejected by a higher-level classifier.

In top-down level-based hierarchical classification, each classifier CF_k associated with an internal category has its own *work domain*, denoted by $WD(CF_k)$, that is a set of categories which documents can only be assigned to after the latter have been accepted by CF_k. CF_k is also known as a *subtree classifier*. For example, all the categories in the category tree belong to the work domain of the classifier at root level $WD(CF_{root})$. Given a real category C_i, the *blocking* of C_i by classifier CF_k ($C_i \in WD(CF_k)$), denoted by $blocking(C_i, CF_k)$, is defined as the number of documents that belong to category C_i but are rejected by CF_k. For example, the category *conference* is a leaf category taken from Yahoo! hierarchy, and the path to reach it is : *Root* > *Computers and Internet* > *Software* > *Programming Tools* > *Object-Oriented Programming* > *conference*; *blocking* ($conference$, $CF_{Software}$) = 3 means that 3 documents that belong to category $conference$ are wrongly rejected by the classifier that accept documents for the category subtree rooted at $Software$.

The *blocking factor* for a classifier CF_k, denoted by $BF(CF_k)$, is defined as the proportion of documents belonging to $WD(CF_k)$ blocked by CF_k.

$$BF(CF_k) = \frac{\sum_{C_i \in WD(CF_k)} |blocking(C_i, CF_k)|}{\sum_{C_i \in WD(CF_k)} |C_i|} \qquad (21)$$

In Equation 21, $|C_i|$ refers to the total number of documents that are labelled with C_i.

The above blocking factor definition is different from the other measures such as precision and recall for evaluating the performance of an entire classification method. Blocking factor is classifier-centric as it is defined for each subtree classifier and applies to top-down level-based hierarchical classification only. By comparing the blocking factors of subtree classifiers, the non-performing ones can be identified. Several strategies can be explored to use the blocking factor to improve a top-down level-based hierarchical classification method. The different strategies for classifier replacement and parameter tuning are topics of our future research.

5 Experiment and Results

In this section, we apply our proposed performance measurement framework to evaluate and compare two hier-

archical classification methods based on top-down level-based approach. We first describe the data set used in our experiments and then present two hierarchical classification methods for category trees based on top-down level-based approach. The methods use SVM classifiers and binary Naïve Bayes classifiers respectively. Dumais and Chen had shown that SVM works well for virtual category tree but they did not consider category tree in their work (Dumais & Chen 2000). On the other hand, Naïve Bayes classifiers are considered as baseline classifiers in text classification. We will compare the classification performance of these two hierarchical classification methods using the proposed performance measures.

5.1 Data Set

In our experiment, Reuters-21578 news collection was used. Reuters corpus is one of the most popular data set used in text classification (Yang 1999). Much work in hierarchical text classification have also used this collection (D'Alessio et al. 2000, Koller & Sahami 1997, Weigend et al. 1999). The 21578 documents in this collection are organized into 135 categories. Each document may have zero, one or more categories labelled to it. The categories are not organized in hierarchical manner. To conduct experiment, category trees need to be manually derived from the 135 categories. Koller and Sahami (Koller & Sahami 1997) extracted three category trees from the Reuters-22173 collection by identifying the category labels that suggest parent-child relationships. We have derived three category trees from the Reuters-21578 collection using a similar approach (see Figure 1).

Almost all documents in Reuters collection come with title, dateline and text body. In our experiment, we used title and text body only. After stopword removal and stemming, a *binary feature vector* was obtained for each document without applying any other feature selection. The stopword list and the stemming algorithm were taken directly from the BOW library (McCallum 1996). In our experiment, we used *Lewis Split* to split the Reuters collection into training set and test set.

5.2 Hierarchical Classification Method Based on Binary Naïve Bayes Classifiers

Naïve Bayes classifier is constructed based on the famous Bayes theorem (Mitchell 1997). In our experiment, we adopted the Bernoulli model and use the binary feature vectors in Section 5.1 as document vectors. To compare the performance with the SVM classifier easier (reported in Section 5.3), a binary Naïve Bayes classifier has been implemented based on the formulas described in (McCallum & Nigam 1998).

As binary classifiers can only make decisions of accepting or rejecting the documents, for each *real category* in our given category trees, Trees (a), (b) and (c), a binary Naïve Bayes classifier is required. These classifiers are known as *local classifiers* and they determine whether documents should be assigned to the corresponding categories. *Subtree classifiers*, on the other hand, determine whether documents should be assigned to category subtrees. Subtree classifiers are assigned to all the internal categories including the root category, such as, *grain*, *crude*, and *Hier1* in Tree (a). Note that only if documents are accepted by the subtree classifier, the local classifiers in the work domain of the subtree classifier will have the chance to further classify these documents. For example, if the subtree classifier at *crude* accept a document, the local classifier for category *crude* will be able to determine whether to accept this document.

Binary classifiers need to be trained with both positive and negative documents associated with the categories the classifiers are assigned to. Given a hierarchy H, the $Coverage$ of category C_i in H, denoted by $Coverage(C_i)$, is the set of *real categories* that belong to the category subtree rooted at C_i (including C_i if it is real). Note that $Coverage(C_i) = \{C_i\}$ if C_i is a leaf category in the category tree since all the leaf categories are real. For any document d_j, $d_j \in C_i$ is true if and only if d_j belongs to category C_i; $d_j \in Coverage(C_i)$ is true if and only if d_j belongs to *any* of the categories in $Coverage(C_i)$. In cases where one training document d_j is labelled with more than one category, $d_j \in C_i$ is true if and only if C_i is one of d_j's labelled categories. Let $Parent(C_i)$ denotes the parent category of C_i. The selection of training documents for each of the classifiers was done as follows, where *Positive* and *Negative* refer to the positive training documents and negative training documents respectively.

- Subtree-classifier for root category C_{root}:
  - Positive: All training documents d_j's such that $d_j \in Coverage(C_{root})$.
  - Negative: Training documents d_j's such that $d_j \notin Coverage(C_{root})$. These are documents belonging to categories outside the category tree. However not all of these documents were selected as negative training documents due to the large number of them. We chose to use equal number of positive and negative documents. The negative documents were therefore randomly selected from the documents that $d_j \notin Coverage(C_{root})$.
- Subtree-classifier at internal category C_i:
  - Positive: All training documents d_j's such that $d_j \in Coverage(C_i)$.
  - Negative: All training documents d_j's such that $d_j \in Coverage(Parent(C_i))$ and $d_j \notin Coverage(C_i)$.
- Local-classifier of an internal category C_i:
  - Positive: All training documents d_j's such that $d_j \in C_i$.

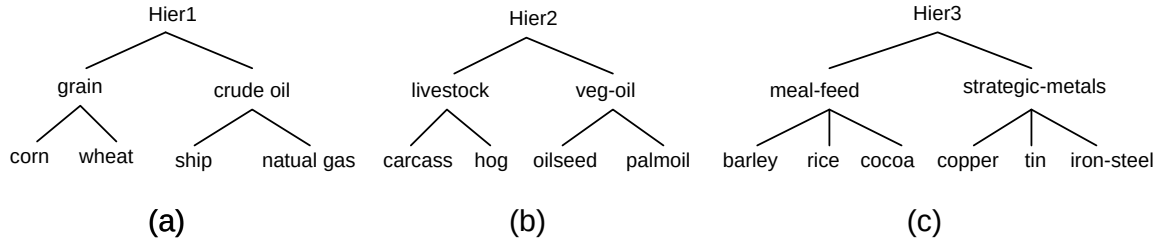

Fig. 1 Hierarchical trees used in our experiment

  - Negative: All training documents d_j's such that $d_j \in Coverage(C_i)$ and $d_j \notin C_i$.
- Local-classifier for a leaf category C_l:
  - Positive: All training documents d_j's such that $d_j \in C_l$.
  - Negative: All training documents d_j's such that $d_j \in Coverage(Parent(C_l))$ and $d_j \notin C_l$.

Note that the root category is a virtual category, and hence is not assigned with a local classifier.

All the test documents that belong to $Coverage(Root)$ in each category tree were selected as its positive test documents. Similar to the training documents, same number of negative test documents were randomly selected from the test documents that not belong to $Coverage(Root)$. The statistics about our training and test documents are shown in Table 2. In the table, $+Tr$, $-Tr$ and Te refer to number of positive training, negative training and test documents respectively. For internal category C_i, $C_i : s$ refers to the category subtree rooted at C_i while $C_i : l$ refers to the category itself.

From the training documents of any two categories, we compute cosine similarity measure as the category similarity. Let each training document be represented by a vector, a super document can be constructed for each category by summing up all the training document vectors. This super document vector is known as *category vector*. The similarity between the two categories can then be computed as the cosine of the two category vectors. It should be noted that the category similarity is independent of the hierarchy given. However, it does depend on how the features are selected for the categories when these features are used to compute the category similarities.

The category similarity matrices for the three category trees were computed and shown in Tables 3, 4 and 5. As the category similarity is based on cosine distance and hence $CS(C_i, C_k) = CS(C_k, C_i)$, only the lower triangular parts are shown in the three tables.

The classification results measured by the standard, category similarity based and distance based precisions and recalls are shown in Table 6. The precisions and recalls are denoted by Pr, Re, Pr^{CS}, Re^{CS}, Pr^{DB} and Re^{DB} respectively. For each kind of measure, we also computed the micro-average and macro-average values. The Pr^{DB} and Re^{DB} were computed with $Dis_\theta = 2$.

Table 6 Results of method using Naïve Bayes classifiers

Category	Pr	Re	Pr^{CS}	Re^{CS}	Pr^{DB}	Re^{DB}
Tree (a)						
crude	0.724	0.793	0.728	0.796	0.776	0.831
grain	0.874	0.885	0.873	0.884	0.863	0.877
nat-gas	0.629	0.566	0.661	0.578	0.807	0.666
ship	0.755	0.797	0.758	0.798	0.797	0.780
corn	0.612	0.678	0.858	0.855	0.786	0.816
wheat	0.684	0.732	0.866	0.927	0.813	0.849
Micro-Ave	0.745	0.787	0.797	0.831	0.807	0.826
Macro-Ave	0.713	0.742	0.791	0.806	0.807	0.803
Tree (b)						
livestock	0.538	0.583	0.737	0.678	0.630	0.527
veg-oil	0.694	0.675	0.744	0.774	0.734	0.824
carcass	0.615	0.444	0.720	0.588	0.352	0.171
hog	0.666	0.666	0.674	0.621	0.818	0.642
oil-seed	0.640	0.872	0.688	0.900	0.635	0.881
palm-oil	0.600	0.600	0.642	0.684	0.695	0.761
Micro-Ave	0.632	0.690	0.709	0.762	0.656	0.695
Macro-Ave	0.625	0.640	0.701	0.707	0.644	0.634
Tree (c)						
meal-feed	0.538	0.736	0.561	0.750	0.685	0.822
str-metal	0.000	0.000	0.000	0.000	0.500	0.043
barley	0.375	0.642	0.436	0.693	0.551	0.750
rice	0.480	0.666	0.554	0.700	0.196	0.476
cocoa	0.560	0.583	0.571	0.593	0.571	0.571
copper	0.687	0.611	0.696	0.618	0.700	0.567
iron-steel	0.142	0.214	0.149	0.222	0.162	0.250
tin	0.750	0.250	0.819	0.384	0.000	0.000
Micro-Ave	0.464	0.507	0.501	0.539	0.444	0.463
Macro-Ave	0.441	0.463	0.462	0.494	0.629	0.378

From the experimental results shown in Table 6, it is clear that the method performed best for Tree (a) in terms of both the standard and extended measures while there was room for improvement for Trees (b) and (c). According to Table 2, Tree (a) received much larger number of training and test documents than Trees (b) and (c). Therefore, the results obtained from Tree (a) should be more representative compared to the other two trees.

The micro-average and macro-average values give an overall picture of how the performance is. In all the trees, micro-average and macro-average values based on category similarity, Pr^{CS} and Re^{CS}, were higher than the standard ones. At the category level, most of the extended precisions and recalls gave larger values except those of the *grain* category. The contributions made by the misclassified documents are given in Table 7. In the contribution table, the number of documents for TP_i, FN_i and FP_i are given and the contributions for FN_i and FP_i are given in the form of $FnCon$ and $FpCon$ respectively. Block(I) refers to the blocking measure of the subtree classifier rooted at the first level categories

Table 2 Number of training and test documents for Trees (a), (b) and (c)

Tree (a)				Tree (b)				Tree (c)			
Category	$+Tr$	$-Tr$	$+Te$	Category	$+Tr$	$-Tr$	$+Te$	Category	$+Tr$	$-Tr$	$+Te$
Hier1:s	981	981	394	Hier2:s	270	270	104	Hier3:s	271	271	123
crude:s	574	435	-	livestock:s	83	188	-	meal-feed:s	153	119	-
crude:l	391	183	189	livestock:l	75	8	24	meal-feed:l	30	123	19
grain:s	437	544	-	veg-oil:s	191	81	-	str-metal:s	119	153	-
grain:l	434	3	149	veg-oil:l	87	104	37	str-metal:l	16	103	11
nat-gas	75	499	30	carcass	50	33	18	barley	37	116	14
ship	198	376	89	hog	16	67	6	rice	55	98	18
corn	182	255	56	oil-seed	124	67	47	cocoa	35	118	24
wheat	212	225	71	palm-oil	30	161	10	copper	47	72	18
-	-	-	-	-	-	-	-	iron-steel	40	79	14
-	-	-	-	-	-	-	-	tin	18	101	12

Table 3 Category similarity matrix for Tree (a)

Category	crude	grain	nat-gas	ship	corn	wheat
crude	1.000	-	-	-	-	-
grain	0.523	1.000	-	-	-	-
nat-gas	0.602	0.453	1.000	-	-	-
ship	0.574	0.556	0.432	1.000	-	-
corn	0.487	0.791	0.463	0.510	1.000	-
wheat	0.497	0.815	0.472	0.513	0.699	1.000
Average Category Similarity				0.559		

Table 4 Category similarity matrix for Tree (b)

Category	livestock	veg-oil	carcass	hog	oil-seed	palm-oil
livestock	1.000	-	-	-	-	-
veg-oil	0.529	1.000	-	-	-	-
carcass	0.844	0.518	1.000	-	-	-
hog	0.538	0.333	0.468	1.000	-	-
oil-seed	0.537	0.655	0.517	0.340	1.000	-
palm-Oil	0.394	0.664	0.386	0.314	0.452	1.000
Average Category Similarity				0.499		

such as *grain* and *crude* and Block(II) gives the blocking measure of subtree classifier at root level. As shown in Table 7, in Tree (a), category *grain* received negative contributions from the documents in both FP_{grain} and FN_{grain}. From the category similarity matrix, we observed that the categories within the subtree rooted at *grain* or *crude* were more similar to each other compared to the categories across the two subtrees. Therefore, the documents misclassified within the subtree contributed positively to the extended measures based on category similarity. However, $|FN_{grain}| = 17$ and 12 of them were blocked by the subtree classifier at root level according to the blocking measure shown in Table 7. These 12 documents had no contribution to the extended performance measures as they are not assigned any categories. The other 5 documents had the chances to be classified under the subtree rooted at *crude* and contributed negatively (i.e., -0.259) to the performance measures for *grain* based on category similarity. Clearly, some of the documents in FP_{grain}, 19 documents in total, came from the other subtree and hence contributed -0.020. On the other hand, the documents in both FN_{wheat} and FP_{wheat} contributed positively, 13.00 and 12.10 respectively, to the category similarity based measures. That is, these documents were either misclassified into or came from nearby categories (e.g., *grain* and *corn*). Unsurprisingly, *strategic-metal* received zero precision and recall due to inadequate number of training and test documents.

The same observation holds for extended precision and recall based on category distance, i.e., most of the extended precision and recall values were higher than the standard ones. For Tree (a), since the acceptable error distance is 2 (i.e., $Dis_\theta = 2$), the documents misclassified within the subtrees rooted at *grain* and *crude* were likely to contribute positively to the extended precision and recall. The negative contributions could only be found in categories *grain* and *ship*. The reason is that the documents were misclassified into the opposite subtree. Two categories, *carcass* and *rice* from Trees (b) and (c) respectively, received significant decrease in extended measures based on category distance compared to the

Table 5 Category similarity matrix for Tree (c)

Category	meal-feed	str-metal	barley	rice	cocoa	copper	tin	iron-steel
meal-feed	1.000	-	-	-	-	-	-	-
str-metal	0.334	1.000	-	-	-	-	-	-
barley	0.488	0.311	1.000	-	-	-	-	-
rice	0.434	0.330	0.381	1.000	-	-	-	-
cocoa	0.419	0.317	0.422	0.430	1.000	-	-	-
copper	0.401	0.426	0.348	0.415	0.368	1.000	-	-
tin	0.389	0.384	0.348	0.435	0.399	0.424	1.000	-
iron-steel	0.378	0.346	0.338	0.532	0.384	0.397	0.416	1.000
Average Category Similarity					0.393			

Table 7 Contributions and blocking measures of method using Naïve Bayes classifiers

	Num. of Docs			CS Contribution		DB Contribution		Blocking measure	
Category	$\|TP\|$	$\|FN\|$	$\|FP\|$	$FnCon$	$FpCon$	$FnCon$	$FpCon$	Block(I)	Block(II)
Tree (a)									
crude	150	39	57	0.369	0.645	5.500	9.500	0	27
grain	132	17	19	-0.259	-0.020	-1.000	-1.500	5	12
nat-gas	17	13	10	0.001	0.870	1.500	4.500	0	8
ship	71	18	23	0.003	0.258	-2.500	4.500	9	4
corn	38	18	24	7.899	14.12	6.000	9.500	0	6
wheat	52	19	7	13.00	12.10	7.000	8.500	1	2
Tree (b)									
livestock	14	10	12	0.687	4.982	-3.000	3.500	8	2
veg-oil	25	12	11	3.438	0.913	5.500	0.000	0	1
carcass	8	10	5	2.294	0.725	-4.500	-0.500	6	1
hog	4	2	2	-0.331	0.155	-0.500	1.000	1	1
oil-seed	41	6	23	1.042	2.718	0.500	-0.500	1	2
palm-oil	6	4	4	0.800	0.140	1.500	0.500	0	0
Tree (c)									
meal-feed	14	5	12	0.111	0.559	1.000	3.500	0	1
str-metal	0	11	1	-0.069	-0.013	0.000	0.500	1	2
barley	9	5	15	0.304	1.301	0.500	4.000	0	1
rice	12	6	13	0.048	1.838	0.500	-7.500	0	3
cocoa	14	10	11	0.151	0.221	-0.500	0.500	2	1
copper	11	7	5	0.088	0.112	-1.000	0.500	2	2
tin	3	11	18	0.037	0.102	0.500	0.000	0	10
iron-steel	3	9	1	1.619	-0.014	-7.000	-1.000	7	0

Table 8 Blocking factors of method using Naïve Bayes classifiers

Tree (a)		Tree (b)		Tree (c)	
$CF_{subtree}$	$BF(CF)$	$CF_{subtree}$	$BF(CF)$	$CF_{subtree}$	$BF(CF)$
Hier1:s	0.101	Hier2:s	0.049	Hier3:s	0.154
crude:s	0.029	livestock:s	0.313	meal-feed:s	0.027
grain:s	0.021	veg-oil:s	0.011	str-metal:l	0.018

standard ones. For *carcass*, 10 documents were wrongly rejected and among them 6 were blocked by the subtree classifier at *livestock* but accepted by the subtree classifier at the root level. Therefore, these 6 documents could be wrongly accepted by the subtree classifier at *veg-oil* and hence contributed negatively. In the category *rice*, 13 documents out of the 25 accepted ones were wrongly accepted. Some of these wrongly accepted documents might come from the categories that are more than 2 links apart, and thus giving a large negative contribution.

Based on the blocking factors shown in Table 8, the subtree classifiers at the root level for both Trees (a) and (c) were identified as the non-performing ones. The subtree classifier at *livestock* also performed poorly due to very few available training and test documents. Blocking at the root level gave little room for the low level classifiers to deliver good classification results. Therefore, in top-down level-based classification methods, the high level classifiers must be carefully designed to avoid blocking. Otherwise, the classifiers at the low level cat-

egory should re-examine the documents rejected by the high level classifiers.

5.3 Hierarchical Classification Method Based on SVM Classifiers

Support Vector Machine classifiers have been shown to be fast and effective in text classification (Dumais, Platt, Heckerman & Sahami 1998, Joachims 1998). SVM is good at finding the best surface that separates the positive and negative training examples at the widest margin (Joachims 1998). Therefore, classifiers based on SVM are binary classifiers and are trained with both positive and negative examples. The SVM classifier used in our experiment was SVM^{light} Version 3.50 (Joachims 2001) implemented by Joachims.

In this experiment, we simply replaced the binary Naïve Bayes classifiers in Section 5.2 with SVM classifiers. All the other settings remained the same. That is, we selected both the training and test documents in the same way as for the binary Naïve Bayes classifiers. All these will ensured us a fair comparison between the two classification methods.

Table 9 Results of method using SVM classifiers

Category	Pr	Re	Pr^{CS}	Re^{CS}	Pr^{DB}	Re^{DB}
Tree (a)						
crude	0.846	0.962	0.849	0.963	0.890	0.964
grain	0.869	0.939	0.869	0.938	0.868	0.932
nat-gas	0.818	0.600	0.833	0.613	0.900	0.714
ship	0.879	0.820	0.879	0.821	0.888	0.843
corn	0.888	0.857	0.944	0.939	0.929	0.913
wheat	0.886	0.986	0.976	0.993	0.942	0.980
Micro-Ave	0.864	0.909	0.883	0.919	0.895	0.922
Macro-Ave	0.864	0.860	0.892	0.878	0.903	0.891
Tree (b)						
livestock	0.629	0.708	0.816	0.771	0.734	0.654
veg-oil	0.878	0.783	0.906	0.845	0.904	0.880
carcass	0.611	0.611	0.778	0.735	0.600	0.473
hog	1.000	0.333	1.000	0.287	1.000	0.416
oil-seed	0.646	0.893	0.688	0.919	0.625	0.901
palm-oil	1.000	0.700	1.000	0.799	1.000	0.850
Micro-Ave	0.710	0.760	0.789	0.815	0.734	0.769
Macro-Ave	0.794	0.671	0.864	0.726	0.810	0.695
Tree (c)						
meal-feed	1.000	0.315	1.000	0.332	1.000	0.368
str-metal	0.000	0.000	0.000	0.000	0.000	0.000
barley	0.923	0.857	0.923	0.857	0.923	0.857
rice	1.000	0.833	1.000	0.833	1.000	0.833
cocoa	1.000	0.500	1.000	0.505	1.000	0.520
copper	0.937	0.833	0.937	0.833	0.937	0.833
iron-steel	0.500	0.428	0.500	0.428	0.500	0.428
tin	1.000	0.250	1.000	0.249	1.000	0.166
Micro-Ave	0.896	0.530	0.896	0.534	0.896	0.534
Macro-Ave	0.795	0.502	0.795	0.505	0.795	0.501

The results of our experiment using SVM classifiers are presented in Table 9. Similarly to the previous experiments, we computed both the standard and extended precisions and recalls. Pr^{DB} and Re^{DB} were computed with $Dis_\theta = 2$ for easy comparison. We also computed both the standard and extended micro and macro averages. It can be seen that the SVM classifier outperformed Naïve Bayes classifier in all the three trees. Most of the precision and recall values were good. This make our experiment to be consistent with the experimental results given by Koller and Sahami (Koller & Sahami 1997) although slightly different collections are used. The extended precision and recall based on category similarity were generally better than the standard ones. Negative contributions occurred again for *grain* in Tree (a), with the same reason discussed in Section 5.2. Most of the extended precision and recall values based on category distance were higher than the standard ones. For Trees (b) and (c), there were several categories (i.e. *hog*, *tin* and *strategic-metal*) trained with documents fewer than 20. Since SVM classifiers require about 20 training documents to yield stable performance (Dumais & Chen 2000), the performance result might not be representative enough for these categories. Nevertheless, even with the consideration of extended precision and recall, our hierarchical classification method based on SVM was clearly superior to that based on Naïve Bayes classifiers.

The blocking measures are shown in Table 10. Similar to the method based on binary Naïve Bayes classifier, most of the documents were blocked by the classifier at the root. However, compared to the blocking measures for binary Naïve Bayes classifiers, the number of blocked documents were significantly reduced, especially in Tree (a). For example, the number of blocked documents for category *crude* by the subtree classifier at root level was reduced to 7 from 27; $blocking(grain, CF_{root})$ was 6 compared to 12. The reduction of root classifier blocking could also be noticed by comparing the blocking factors for all the Trees. For example, in Tree(a), the blocking factors of the classifier at root level were 0.043 and 0.101 for methods based on SVM and Naïve Bayes classifiers respectively.

6 Conclusions

In this paper, we give an overview of hierarchical classification problem and its solutions. While the commonly-used performance measurement framework for flat classification can be applied to hierarchical classification, the relationships among categories in the category structure are not accounted for. In this paper, we therefore proposed a new performance measurement framework for hierarchical classification. In this framework, we incorporate the contributions of misclassified documents into the definition of performance measures. Furthermore, we developed a new measure known as blocking measure to identify the non-performing classifiers in the classification method. Two top-down level-based hierarchical classification methods based on binary Naïve Bayes classifiers and SVM classifiers have been presented. Using the framework, we evaluated the performance of the two hierarchical classification methods using the Reuters collection. We showed that the one using SVM classifiers performed well when given enough training documents,

Table 10 Contributions and blocking measures of method using SVM classifiers

	Num. of Docs			CS Contribution		DB Contribution		Blocking measure	
Category	$\|TP\|$	$\|FN\|$	$\|FP\|$	$FnCon$	$FpCon$	$FnCon$	$FpCon$	Block(I)	Block(II)
Tree (a)									
crude	182	7	33	0.000	0.745	0.000	9.500	0	7
grain	140	9	21	-0.095	0.000	-1.000	0.000	3	6
nat-gas	18	12	4	0.291	0.290	3.000	1.500	0	5
ship	73	16	10	0.144	-0.009	2.000	0.500	3	5
corn	48	8	6	4.426	2.730	3.000	2.000	0	2
wheat	70	1	9	0.477	7.130	-0.500	4.500	0	0
Tree (b)									
livestock	17	7	10	0.374	4.982	-2.500	3.500	5	2
veg-oil	29	8	4	2.175	0.718	3.500	0.500	0	1
carcass	11	7	7	1.522	2.669	-3.000	1.000	4	1
hog	2	4	0	-0.274	0.000	0.500	0.000	0	1
oil-seed	42	5	23	1.042	2.413	0.500	-1.500	1	1
palm-oil	7	3	0	0.990	0.000	1.500	0.000	0	0
Tree (c)									
meal-feed	6	13	0	0.315	0.000	1.000	0.000	0	0
str-metal	0	11	0	0.000	0.000	0.000	0.000	1	5
barley	12	2	1	0.000	0.000	0.000	0.000	0	1
rice	15	3	0	0.000	0.000	0.000	0.000	0	1
cocoa	12	12	0	0.142	0.000	0.500	0.000	0	0
copper	15	3	1	0.000	0.006	0.000	0.000	1	1
tin	6	8	6	0.000	0.000	0.000	0.000	1	5
iron-steel	3	9	0	-0.007	0.000	-1.000	0.000	2	1

Table 11 Blocking factors of method using SVM classifiers

Tree (a)		Tree (b)		Tree (c)	
$CF_{subtree}$	$BF(CF)$	$CF_{subtree}$	$BF(CF)$	$CF_{subtree}$	$BF(CF)$
Hier1	0.043	Hier2	0.042	Hier3	0.108
crude:s	0.001	livestock:s	0.187	meal-feed:s	0.000
grain:s	0.004	veg-oil:s	0.011	str-metal:l	0.091

while the hierarchical classification method based on binary Naïve Bayes classifiers was less effective. We have also shown that extended performance measures can indeed help us to better determine the performance of hierarchical classification methods by considering contributions from misclassified documents.

In this paper, we have derived the extended performance measures using category similarity and distance that capture the relationships between categories. In our future work, we plan to design new hierarchical classification methods that exploit such information and deliver good classification performance. Using top-down level-based approach, it was mentioned that the blocking error made at the parent category is not recoverable at the child category. In Dumais and Chen's work (Dumais & Chen 2000), a method that can make decisions based on both the results from the parent classifier and child classifier was introduced to overcome this problem. However, no outstanding results were obtained. In our future work, we will also try to design a tolerant hierarchical classification method that allows child classifiers to recover errors made by the parent classifiers. One possible approach is to classify documents based on the outputs of classifiers at current level and all ancestral levels using some voting algorithm.

As much of the information today can be accessed from World Wide Web, the problem of classifying web pages into categories is becoming very important. To conduct hierarchical classification on web pages, the existing classification methods have to be further extended. In addition to text found in the web pages, features should be derived from web page structures and links between web pages as they often carry some useful information about the web pages. Some research on hierarchical web page classification has been reported in (Chakrabarti, Dom & Indyk 1998, Dumais & Chen 2000). We will continue to explore the use of extended performance measures in hierarchical web page classification, and develop methods that use a variety of web page features to achieve better classification results.

References

Chakrabarti, S., Dom, B. E. & Indyk, P. (1998), Enhanced hypertext categorization using hyperlinks, *in* 'Proc. of

the ACM Int. Conf. on Management of Data (SIGMOD98)', Seattle, USA, pp. 307–318.

Cohen, W. W. (1995), Fast effective rule induction, *in* 'International Conference on Machine Learning', pp. 115–123.

D'Alessio, S., Murray, K., Schiaffino, R. & Kershenbaum, A. (2000), The effect of using hierarchical classifiers in text categorization, *in* 'Proc. of the 6th Int. Conf. "Recherche d'Information Assistee par Ordinateur"', Paris, FR, pp. 302–313.

Dumais, S. T. & Chen, H. (2000), Hierarchical classification of Web content, *in* 'Proc. of the 23rd ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR)', Athens, GR, pp. 256–263.

Dumais, S. T., Platt, J., Heckerman, D. & Sahami, M. (1998), Inductive learning algorithms and representations for text categorization, *in* 'Proc. of the 7th Int. Conf. on Information and Knowledge Management', pp. 148–155.

Gaussier, E., Goutte, C., Popat, K. & Chen, F. (2002), A hierarchical model for clustering and categorising documents, *in* 'Proc. of the 24th BCS-IRSG European Colloquium on Information Retrieval Research', Glasgow, UK, pp. 229–247.

Greiner, R., Grove, A. & Schuurmans, D. (1997), 'On learning hierarchical classifications'. http://citeseer.nj.nec.com/article/greiner97learning.html.

Joachims, T. (1998), Text categorization with support vector machines: learning with many relevant features, *in* 'Proc. of the 10th European Conf. on Machine Learning', Chemnitz, DE, pp. 137–142.

Joachims, T. (2001), SVM^{light}, An implementation of Support Vector Machines (SVMs) in C. http://svmlight.joachims.org/.

Koller, D. & Sahami, M. (1997), Hierarchically classifying documents using very few words, *in* 'Proc. of the 14th Int. Conf. on Machine Learning', Nashville, US, pp. 170–178.

Labrou, Y. & Finin, T. W. (1999), Yahoo! as an ontology: Using Yahoo! categories to describe documents, *in* 'Proc. of the 8th Int. Conf. on Information Knowledge Management', Kansas City, MO, pp. 180–187.

Larkey, L. S. & Croft, W. B. (1996), Combining classifiers in text categorization, *in* 'Proc. of the 19th ACM Int. Conf. on Research and Development in Information Retrieval', Zürich, CH, pp. 289–297.

Lewis, D. D. (1992), An evaluation of phrasal and clustered representations on a text categorization task, *in* 'Proc. of the 15th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval', Kobenhavn, DK, pp. 37–50.

Lewis, D. D. (1995), Evaluating and optmizing autonomous text classification systems, *in* 'Proc. of the 18th ACM Int. Conf. on Research and Development in Information Retrieval', Seattle, US, pp. 246–254.

McCallum, A. K. (1996), Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/∼mccallum/bow.

McCallum, A. K. & Nigam, K. (1998), A comparison of event models for Naïve Bayes text classification, *in* 'Proc. of the Workshop on Text Categorization (AAAI98)', Madison, WI, pp. 41–48.

McCallum, A. K., Rosenfeld, R., Mitchell, T. M. & Ng, A. Y. (1998), Improving text classification by shrinkage in a hierarchy of classes, *in* 'Proc. of the 15th Int. Conf. on Machine Learning', Madison, US, pp. 359–367.

Mitchell, T. M. (1997), *Machine learning*, International edn, McGraw-Hill, New York.

Mladenic, D. (1998), Turning Yahoo to automatic web-page classifier, *in* 'Proc. of the 13th European Conf. on Artificial Intelligence', Brighton, UK, pp. 473–474.

Rijsbergen, C. J. V. (1979), *Information Retrievel*, 2nd edn, London: Butterworths.

Robertson, S. & Hull, D. A. (2000), The TREC-9 filtering track final report, *in* 'Proc. of the 9th Text REtrieval Conference (TREC-9)', Gaithersburg, Maryland.

Sasaki, M. & Kita, K. (1998), Rule-based text categorization using hierarchical categories, *in* 'Proc. of the IEEE Int. Conf. on Systems, Man, and Cybernetics', La Jolla, US, pp. 2827–2830.

Sebastiani, F. (2002), 'Machine learning in automated text categorization', *ACM Computing Surveys* **34**(1), 1–47.

Toutanova, K., Chen, F., Popat, K. & Hofmann, T. (2001), Text classification in a hierarchical mixture model for small training sets, *in* 'Proc. of the 10th Int. Conf. on Information and Knowledge Management', Atlanta, USA, pp. 105–112.

Vinokourov, A. & Girolami, M. (2002), 'A probabilistic framework for the hierarchic organization and classification of document collections', *Journal of Intelligent Information Systems* **18**(2/3), 153 – 172. Special Issue on Automated Text Categorisation.

Wang, K., Zhou, S. & He, Y. (2001), Hierarchical classification of real life documents, *in* 'Proc. of the 1st SIAM Int. Conf. on Data Mining', Chicago, USA.

Wang, K., Zhou, S. & Liew, S. C. (1999), Building hierarchical classifiers using class proximity, *in* 'Proc. of the 25th Int. Conf. on Very Large Data Bases', Edinburgh, UK, pp. 363–374.

Weigend, A. S., Wiener, E. D. & Pedersen, J. O. (1999), 'Exploiting hierarchy in text categorization', *Information Retrieval* **1**(3), 193–216.

Yang, Y. (1999), 'An evaluation of statistical approaches to text categorization', *Information Retrieval* **1**(1-2), 69–90.