On Using Hierarchies for Document Classification

Wahyu Wibowo and Hugh E. Williams

Department of Computer Science
RMIT University
GPO Box 2476V, Melbourne 3001, Australia
{*wwibowo,hugh*} @*cs.rmit.edu.au*

Abstract

Good management of large collections, such as world-wide web databases or newswire services, is essential to ensure that they remain useful resources. Large collection management tasks include storing, querying, retrieving, routing, filtering, and classifying documents. We focus in this paper on new approaches to the last of these tasks, classification. Classification is the process of assigning one or more identifiers from a list of classes to a document. The identifier or class label is useful to organise, retrieve, or present documents. Several factors affect the effectiveness of classification schemes, including the classification method, selection of training samples, selection of features, and class label assignment methods. We identify problems in classification, propose a new evaluation framework, and show that using hierarchical information, where parent classes and subclasses of labels are used, has potential to improve classification effectiveness.

Keywords Document Management, Document Databases, Document Classification, Information Retrieval, SGML and Markup.

1 Introduction

Large online collections continue to grow in volume and to place demands both on space resources and on techniques to manage and query databases. For querying of world-wide web databases, Boolean and ranked querying remain the most popular techniques for finding relevant documents to our information needs. However, alternative techniques that offer different ways of exploring information, such as the Yahoo! search engine [1] hierarchical approach or browsing with phrases [15], are also proving useful methods to satisfy our search needs.

Text classification, where documents are assigned one or more labels from a predefined set, is one possible technique that can be used to improve the organisation and management of data. In the case of Yahoo! and in many specialist domains—such as keyword assignment in the Medline bibliographic database [5] or in the GenBank genomic database [2]—such classification is a manual process, where trained experts assign new documents to classes and organise classes into a hierarchy where each node can potentially have multiple parents and children. Building such knowledge bases, while successful and popular for specialist applications, requires significant effort in development.

Automatic class label assignment, without the need for human intervention in the classification process, is desirable for speed, scalability, and cost. However, automatic classification is an uncertain and difficult process. Automatic techniques to accurately classify documents must identify the characteristics of documents that belong to a certain class, suppress noise that may affect the assignment judgement, and be able to determine the class or classes of a document based on the available data.

In automatic classification *training* with sample documents is used to develop a *classifier*. The sample documents in this approach are manually annotated with class labels and the classifier trained to identify the characteristics of each document class. After the training process is complete, unclassified documents are processed and compared to the statistics and characteristics of the training documents to identify features and assign class labels. Automatic classification, in contrast to manual classification, has not been used to classify documents into a hierarchy, but has only been used to classify documents into one or more classes from a set of classes.

We experiment in this paper with classification of documents based on a training set derived from the Reuters newswire service and consider whether the effectiveness of this classification approach can be improved by considering the relationships between different class labels. Specifically, we extend previous experiments and propose a new measurement framework to show that by arranging the class labels in a *hierarchy,* where a document can

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

Proceedings of the Fourth Australasian Document Computing Symposium, Coffs Harbour, Australia, December 3, 1999.

be classified into a tree structure of parents and children, that the accuracy of classification may be able to be improved by considering the relationship between parent and child nodes in the class label tree structure. Our conclusion is that classification of documents into broader parent classes is more accurate than classification into child classes. We expect that hierarchical classification, where parent classification information is used to aid child classification, will improve the accuracy of automatic document classifiers.

2 Document Classification

Development of a trained classifier for automatic classification of documents first requires a training technique based on the pre-classified documents. In deciding how to train the classifier, two questions must be addressed: first, what features of the documents will be derived and used to train the classifier to recognise documents; and, second, how will features be represented and used in the classifier? We discuss these two questions in this section and begin by considering techniques for feature extraction from training documents.

In document retrieval systems, one common way to represent a document is by viewing the document as a collection of features in the form of *words* or *terms,* that is, as unit strings of characters in a document that are separated by white space characters. Using terms, a document can then be represented as a feature space using a vector model $(i_1, i_2, \cdots, i_n)$ where each element i_j is either 0 or 1 depending upon the existence of a term in a document, the occurrence frequency of a term in the document, or *weights* that reflects the importance of an index term. Other approaches to representing documents include using structure or mark-up information to represent a document, but we do not discuss these here.

Many schemes have been proposed for weighting terms within a document, that is, representing the significance of a term in a document in such a feature space model. Most schemes are based on variations of the TF.IDF [11] measure in which the importance of a term in a document is the product of the frequency of the term in the document and the inverse of the number of documents that contain this term. In this way TF.IDF is often calculated as

$$\text{TF.IDF}(i,j) = \text{tf}_{i,j} \times \log(\frac{N}{\text{df}_i})$$

where $\text{tf}_{i,j}$ is the term frequency of term i in document j, df_i is the number of documents that contain term i, and N is the number of documents in the document set.

This approach of representing a document can be generalised to a scheme for representing a class of documents that have the same features. By selecting all training documents that are pre-assigned to a class, it is possible to represent the terms in a class of documents as a vector $c_i = (td_{i,1}, td_{i,2}.., td_{i,n})$ where c_i is the represented document, $td_{i,j}$ is the j-th term descriptor in document i, and n is the number of term descriptors. This vector then is a descriptor of features of a specific class and, as we discuss later, comparison of unclassified documents to such a vector can be used for classification. These approaches to *linear classification* have been shown elsewhere to be an effective method of developing class features from training sets [1, 7, 9]; we describe details of one of these approaches, that of Lam and Ho [7], in the next section.

In representing documents as a vector, some words or terms may be *stopped* in the document identification process [16]. These words are usually common words used by almost all type of documents such as articles (for example "a" or "the"), prepositions (for example "to", "for", or "at"), or very common words (for example "while", "if, "else", "before", or "after"). In addition to consuming processing time in classification, when such terms are removed documents may be more separable for class identification. The terms are eliminated using a static list of stop words, or using a feature selection algorithm such as document-frequency thresholding, information gain criteria, term-strength criterion, mutual information, and χ^2-Tests [17]. The results presented in this paper use simple document-frequency thresholding to remove common terms.

2.1 Rocchio Classifier

The approach to representing a class of documents as a vector, as described in the last section, is both simple and practical. However, this scheme only considers positive information, where the presence of a term in a document class adds that term to the vector. Another approach is to consider negative information, where the presence of a term in a different class reduces the weight or importance of a term. In this way, all terms in the collection are represented in a class vector, with terms present in the class typically having positive weights and those not present in the class typically having negative weights.

Several possible candidate schemes exist for feature representation that includes negative and positive weights [7, 9, 10, 14]. We use the Rocchio weight learning technique [10] which, while having been shown to be marginally less effective for classification than other approaches [7], is simple to implement and practical for our experiments in studying hierarchical classification techniques.

The Rocchio approach is based on a simple similarity measure, where the similarity of two vectors is computed; in this case, the two vectors are for a new training document and an existing vector that represents a class. Such *linear similarity* is computed with a dot product of the two vectors w, the weight vector for a class, and x, the vector of the training document, so that:

$$f(x) = w \bullet X = \sum_{j=1}^{d} w_j x_j$$

where d is the number of term descriptors in vectors w and x.

The Rocchio measure uses the linear similarity measure as follows: for each new training document, the class representative vector for each class is modified by adding the weight of the linear similarity of the positive training terms and subtracting the weight of the linear similarity of the negative training terms. For a class representative vector w and a new training document x, the Rocchio measure is:

$$new\ w_j = \alpha w_j + (3 \frac{\sum_{\in C} x_{i,j}}{|C|} - 7 \frac{\sum_{i \notin C} x_{i,j}}{n-\backslash C\backslash}$$

where n is the total number of training examples, C is the set of positive training examples, and α, β, and γ are constants. In our experiments, we use constants of $\alpha = 1$, $\beta = 16$, and 7 = 4, the same as those reported by Lewis et al. [9].

Given the Rocchio measure, it is then a simple batch process to derive a class feature vector for a given set of classes and training documents. For each training document, the weight of the vector of each class is modified so that when a document is a class member the weight of class terms are increased and when a document is not a class member the weight of non-class terms is decreased. The result is a vector for each class w of length t, where t is the count of distinct, unstopped terms in the collection; in this way, all class vectors are of length t and contain all unstopped terms.

3 Training Documents

Given a Rocchio classifier based on the TF.IDF weighting scheme, as described in the last section, we need to consider suitable test collections, how classes are trained, and how unclassified documents will be compared to the class feature vectors by the classifier. Most importantly, we need to consider how to measure when a test document is assigned or not assigned to a class.

3.1 Reuters Test Collection

Several test collections have been used as training and test sets for linear text classifiers, including the AAP Newswire [3, 8], the MEDLINE Database [17], the Ohsumed database [7], and the Reuters Collection [3, 6, 7, 18]. In our experiments, we use the Reuters-21578 text categorisation[2] test collection of Reuters newswires from 1987 to 1991[3]. This collection contains 21,578 SGML articles stored in 22 data files. Each article in the collection is headed with a tag of the form:

<REUTERS TOPICS=?? LEWISSPLIT=??
CGISPLIT=?? OLDID=?? NEWID=??>

Some documents in the Reuters-21578 collection have no class assignment information in the article header, while others contain irrelevant information. Such documents will not be useful for classification experiments and, because of this, several different splits or divisions of this collection have been proposed for research experiments. Each division is based on the value of the starting tags and we follow the approach of deriving a split between training and test documents proposed by Apte et al. [1], the so-called "Mod-Apte" division.

In "Mod-Apte", the Reuters-21578 collection is divided into three sets:

1. "Training Set": 9,603 documents with the following tags: LEWISSPLIT="TRAIN"; TOPICS="YES"

2. "Test Set": 3,299 documents with the following tags: LEWISSPLIT="TEST"; TOPICS="YES"

3. "Unused": 8,676 documents with the following tags: LEWISSPLIT="NOT-USED"; and either:
   - TOPICS="YES" or
   - TOPICS="NO" or
   - TOPICS="BYPASS"

The documents in the training and test sets in the "Mod-Apte" divisions contain class assignments, where the class assignments are hierarchical. At the *parent* level, there are six classes: "COMPANIES" (which has no positive training examples), "TOPICS" (7,775 positive examples), "PLACES" (8,959 positive examples), "ORGS" (456 positive examples), "PEOPLE" (433 examples), and "EXCHANGES" (73 positive samples). At the *child* level, where each child is associated with one parent class, there are 368 classes, including topics as diverse as "ALGERIA", "AMEX", and "ALUM(inium)". Several child classes have more than 1,000 positive training examples, while 80 classes have only one positive training example. Note that documents can be

[2] The term "categorisation" is sometimes used to describe the process of assigning predefined class labels to documents. We use the term "classification" throughout to describe this process.

[3] The Reuters-21578 collection is available at `http://www.research.att.com/~lewis`

classified into multiple parent and child classes and can therefore be positive examples for multiple classes, as well as always being negative examples for the remaining classes.

3.2 Training the Classifier

In previous experiments [7], linear classification has focused on evaluation with the children of one parent class, for example "TOPICS", using only child classes with more than one positive training example. Our focus in this paper is expanding this approach to investigate the classification of documents into both parent and child classes. In training our classifier, we use all 9,603 documents in the "Mod-Apte" training set and generate class feature vectors for each of the parent classes and child classes, giving a total of 374 vectors each with a length or number of weighted terms of just over 20,000. In our experiments, which we describe in the next section, we compare each of the 3,299 test documents to each of the 374 feature vectors.

3.3 Classifying Documents

Given a trained classifier and a set of test documents, the question remains as to how we decide how does a document fit in a hierarchy of classes and can hierarchy information be used to better classify documents? A simple approach, and the one we employ, is to quantify classification by calculating a vector for each test document using TF.IDF and then calculate the dot product of the test document vector and each of the parent and child feature class vectors. The results, which are similarity scores between the test document and the parent class feature vectors, and similarity scores of the test document to the child class vectors, can then be ranked in order of decreasing similarity. In our case, in contrast to previous approaches, further assessment of the accuracy of these rankings is a particularly difficult problem, as each document can be specified as belonging to multiple parent and child classes.

Several different approaches have been proposed for quantifying the accuracy of rankings of linear classification schemes [18]. One of the more popular approaches is to calculate a "breakeven recall-precision point", where: the number of true positive classifications, true negatives, false positives, and false negatives are summed; standard recall and precision calculated; and, values interpolated to give a breakeven point (a point where recall and precision are the same) [7]. This "breakeven point" approach has been criticised as being artificial [12], since the point calculated is interpolated and represents a point not achievable by the system.

We propose a new technique for evaluating the performance of classification into the parent-child hierarchy of the Reuters-21578 collection by using the number of expected answers as a cut-off point for assessment, a similar approach to that of R-precision [4] or missed-at-equivalence [13]. In this new approach, the number of expected answers is the number of parent or child classes that a test document is assigned to. We use this number as a cut-off point for measuring the number of correct and incorrect assignments.

To illustrate this approach, consider the ranking of a test document against the feature parent classes, where the test document has been manually assigned by a human judge to two parents "PLACES" and "TOPICS". After the comparison with the Rocchio classifier, the similarity to the feature class vectors returned is ranked in the order: "PLACES", "PEOPLE", and then "TOPICS". As the number of correct parents is two, we use a cut-off of two to assess the parent answers; in this case, the first two answers contain one correct parent response "PLACES" and one incorrect parent response "PEOPLE". Similarly, for the children the manual assignment is of eight classes: "trade" (a child of "TOPICS"), "malaysia" (PLACES), "south-korea" (PLACES), "australia" (PLACES), "hong-kong" (PLACES), "usa" (PLACES), and "japan" (PLACES).

We combine our cut-off assessment of parents with a cut-off assessment of child rankings. In the left-most column of Table 1, a ranking is shown of the similarity of the test document to the child feature classes using our classifier; the parent class of each of the ranked results is shown in the middle-column. As with the parent assignments, we cut-off the list at the number of correct child assignments.

The right-most column of Table 1 shows the results of our assessment using this cut-off approach. For each answer, we assess whether the parent of the child was correctly identified above the cut-off—if the parent was identified we say that "P=T(rue)" and otherwise "P=F(alse)". We also assess whether the child answer was correctly identified above the cut-off by indicating "C=T(rue)" or "C=F(alse)". After assigning a value to "P" and "C" for each child above the cut-off, we sum the results of each of the four combinations of "P" and "C" values.

In our example in Table 1, there are five "P=T C=T", zero "P=T C=F", one "P=F C=T", and two "P=F C=F". An assignment of "P=T C=T" is a correct assignment: we have successfully identified a child class above the cut-off and also correctly identified the parent class of that child above the cut-off. The opposite case is "P=F C=F", where the child is incorrectly identified and the parent is also not correctly identified (either a false positive or false negative). The two remaining cases are partially correct identification. The first case is a correct parent, where we identify an incorrect child, that is, the identified child is a sibling of a correct child. The second case is a correct child, where we have not correctly identified its parent.

Ranked Child Results	Parent	Assessment
trade	TOPICS	P=F, C=T
japan	PLACES	P=T, C=T
hong-kong	PLACES	P=T, C=T
south-korea	PLACES	P=T, C=T
taiwan	PLACES	P=T, C=T
gatt	ORGS	P=F, C=F
usa	PLACES	P=T, C=T
yeutter	PEOPLE	P=F, C=F

Table 1: *Sample ranking of child feature classes for a test document. Eight child assignments were made in a manual assessment and we show the top eight ranked reponses from the classification process. The parent of each ranked child is shown in the middle column. The third column shows an assessment: "P=T(rue)" when we have identified the parent of this child in our parent ranking and this agrees with the manual assignment; "C=T(rue)" when we have correctly identified a child matching the manual assessment. The ranking of parents has identified one of two correct parent classes, "PLACES", but not identified the class "TOPICS".*

	Correct Child	Incorrect Child
Correct Parent	5,973	1,484
Incorrect Parent	126	486

Table 2: *Summed classifications of test documents into a hierarchy derived from the Reuters-21578 test collection using 3,299 test documents on a trained set of 6 parent classes and 368 child classes. "Correct" means that the class was ranked above the cut-off and incorrect indicates the class was not ranked above the cutoff.*

4 Hierarchical Classification

Using our approach of analysing the results of hierarchical classification, it is possible to quantify and analyse the likelihood of hierarchies improving the accuracy of classification. The results of our analysis of child-parent classification is shown in Table 2.

Our results shown in Table 2 show the process of testing 3,299 documents against the hierarchy of 6 parent classes and 368 child classes. In these tests, 8,069 manual class assignments were made by human assessors, an average of 2.4 parent and child classes per document. Of these manual class assignments, 74% were classified correctly by our Rocchio classifier both into the correct parent and correct child class, that is, we had identified a child and also correctly identified its parent.

The cases of most interest in our hierarchy are those where either or both the parent and child classification fails. Around 6% of classifications were complete failures, where a child of an unrelated parent was identified. Around 18% were sibling identifications, where an incorrect child is identified of a correct parent, and the remaining around 1% were cases where we identified a correct child but had not identified its parent.

Our results show that when classification fails, it most often fails in child classification. Indeed, in our experiments identification of a sibling is more than 11 times more likely than incorrect parent identification. This is not surprising, since parent classes represent broader topic areas than child classes and we therefore expect that parent vectors are more likely to be separable and distinct than are child vectors.

The skewing of classification failure suggests that parent classes can be used to improve the classification into child classes. If a parent class is much more likely to be identified correctly than a child class, then the parent information can be used to adjust the rankings of children to prefer children that are members of the parent class. Such an approach will lower the ranking of incorrect children, improving the rankings of both correct children and siblings, and reducing both the total failure and the correct parent/incorrect child figures.

Note that we have also observed, as have others [7], that classification failure is more likely when few training documents are available. As an example, many incorrect parent assignments are made to the class "COMPANIES", which has no positive training documents, and many incorrect child assignments are made to classes with one or a only a few positive training examples.

5 Conclusion

Classification of documents into classes is one technique to improve the management of large collections to ensure they remain useful resources. Indeed, document classification by assignment of identifiers from a predefined list is a valuable tool to organise, retrieve, or present documents. Such classification is generally performed with simple linear classifiers, where documents are classified into one or more classes by a pre-trained classifier.

We have presented in this paper a new framework for evaluating classification, which is based on analysis of classification of documents into hierarchies. We have identified problems in classification and shown that using hierarchical information, where parent classes and subclasses of labels are used, has potential to improve classification effectiveness. Indeed, while the assignment process is somewhat affected by the training sets available, the simple ranking algorithm used, and the use of simple Rocchio, our new framework focuses on the relative success of classification into parent and child classes.

Our results indicate that parent classification is more likely to be successful than child classification because of parent classes covering broader, more distinct topic areas. By using parent classifications, which are more likely to be correct, it is likely that hierarchies may be an aid to improving classification in the more fine-grain child classes and removing false-positives. We are currently developing new techniques based on this analysis of classification into hierarchies and believe that hierarchical classification is a valuable tool in improving the accuracy of classification techniques.

Acknowledgments

We thank Ross Wilkinson for valuable discussions and the referees for their feedback. This work was supported by the Multimedia Database Systems group at RMIT University.

References

[1] C. Apte, F. Damerau and S. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems,* Volume 12, Number 3, pages 233-251, 1994.

[2] D.A. Benson, M.S. Boguski, D.J. Lipman, J. Ostell and B.F. Ouellette. GenBank. *Nucleic Acids Research,* Volume 26, Number 1, pages 1-7, 1998.

[3] W.W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. pages 298-306, New York, August 18-22 1996. ACM Press.

[4] D. Harman. Overview of the second text retrieval conference (TREC-2). *Information Processing & Management,* Volume 31, Number 3, pages 271-289, 1995.

[5] W.R. Hersh and R.B. Haynes. Evaluation of SAPHIRE: an automated approach to indexing and retrieving medical literature. In *Proceedings of the 15th Annual Symposium on Computer Applications in Medical Care,* pages 808-812, 1991.

[6] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning (ECML),* 1998.

[7] W. Lam and C.Y. Ho. Using a generalized instance set for automatic text categorization. In R. Wilkinson, B. Croft, K. van Rijsbergen, A. Moffat and J. Zobel (editors), *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval,* pages 81-89, Melbourne, Australia, July 1998.

[8] D.D. Lewis and W.A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and C.J. van Rijsbergen (editors), *SIGIR 94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval,* pages 13-12, London, 1994. Springer-Verlag.

[9] D.D. Lewis, R.E. Schapire, J.P. Callan and R. Papka. Training algorithms for linear text classifiers. In Hans-Peter Frei, Donna Harman, Peter Schäuble and Ross Wilkinson (editors), *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval,* pages 298-306, New York, August 18-22 1996. ACM Press.

[10] J.J. Rocchio. Relevance feedback in information retrieval. In *The Smart Retrieval System — Experiments in Automatic Document Processing,* pages 313-323. Prentice-Hall, Englewood, Cliffs, New Jersey, 1971.

[11] G. Salton (editor). *The SMART Retrieval System—Experiments in Automatic Document Processing.* Prentice-Hall, New Jersey, 1971.

[12] R.D. Schapire, Y. Singer and A. Singhal. Boosting and rocchio applied to text filtering. In R. Wilkinson, B. Croft, K. van Rijsbergen, A. Moffat and J. Zobel (editors), *Proc. ACM-SIGIR International Conference on Research and Development in Information Retrieval,* pages 215-223, Melbourne, Australia, July 1998.

[13] E.G. Shpaer, M. Robinson, D. Yee, J.D. Candlin, R. Mines and T. Hunkapiller. Sensitivity and selectivity in protein similarity searches: A comparison of Smith-Waterman in hardware to BLAST and FASTA. *Genomics,* Volume 38, pages 179-191, 1996.

[14] B. Widrow and S.D. Stearns. *Adaptive Signal Processing.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[15] H.E. Williams, J. Zobel and P. Anderson. What's next? Index structures for efficient phrase querying. In John Roddick (editor), *Proc. Australasian Database Conference,* pages 141-152, Auckland, New Zealand, January 1999.

[16] I.H. Witten, A. Moffat and T.C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images.* Van Nostrand Reinhold, New York, 1994.

[17] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In W. Bruce Croft and C.J. van Rijsbergen (editors), *SIGIR 94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval,* pages 13-22, London, 1994. Springer-Verlag.

[18] Y. Yang. An evaluation of statistical approaches to text categorization. Technical report, CMU-CS-97-127, Carnegie Mellon University, 1997.