Document Classification via Structure Synopses

Liping Ma **John Shepherd** **Anh Nguyen**

School of Computer Science and Engineering
University of New South Wales
Sydney, NSW 2052, Australia
Email: {mliping,jas,anht}@cse.unsw.edu.au

Abstract

Information available in the Internet is frequently supplied simply as plain ascii text, structured according to orthographic and semantic conventions. Traditional document classification is typically formulated as a learning problem where each instance is a whole document that is represented by a feature vector. Such feature vectors are often generated based on the appearance and frequencies of words in the documents. The high-dimensionality of these feature vectors causes some problems: important clues might be missed out, and the classification might be misled by some trivial elements. In this paper, we propose a method which makes use of structuring conventions to reduce size of the feature vector without affecting the accuracy of the classification process. Effectively, a synopsis of document structure is extracted, which contains only the most informative features; then a succinct feature vector is generated to represent the instance. Finally, a decision tree machine learning algorithm is used to classify the document based on its succinct feature vector.

Keywords: Text Classification, Semistructured Data, Information Extraction

1 Introduction

The popularity of the Internet and World Wide Web has increased the need for information management of electronic texts. Text classification aims to automatically categorise text documents into pre-defined classes or types based on their contents. The field of information extraction, on the other hand, tackles the problem of extracting relevant information from this textual data. However, we believe that information extraction can aid text classification by identifying a small set of features in each document that provide very effective discrimination for classification.

The standard document representation used in text classification is the vector space model. In this model, each document is represented by a vector of (*feature*, *value*) pairs. Features are textual units such as words or phrases, also called *terms*. Values can be the presence, the frequencies, or the weights of terms. A collection of documents can then be represented as a set of document vectors or, alternatively, as a matrix $\mathcal{D} = \{v_{ij}\}$ where v_{ij} is the corresponding feature value of term t_i in document d_j.

Since the total number of words in the document collection is large and each individual word may not appear in every document, the matrix $\mathcal{D}$ is often sparse. Despite this, the computation of the classification process can be extremely costly. Moreover, this model does not take advantage of informative terms. A common way to rank the importance of a term is based on its frequency, so that frequent terms may be given higher weights and/or less frequent terms may be discarded. This approach is subjective since terms with high frequency are not necessarily important. Applying the second cutoff can reduce the size of feature matrices but may miss out important terms as its consequence. Another problem in many systems using vector space model is that if the training data is insufficient, the classification tends to become unreliable, i.e. the accuracy will be low. Therefore, improvement can possibly be achieved by not only reducing the size of original feature sets but together with increasing the quality of features.

In dealing with the problems caused by the high dimensionality of feature vectors, two major classes of dimension reduction methods have been developed: feature selection and re-parameterisation (K.Aas & L.Eikvil 1999). Feature selection methods aim to select the most informative words in order to improve the classification effectiveness and reduce the computational complexity. Re-parameterisation methods combine and transform the original features to produce new feature vectors that have smaller dimension.

A large number of useful online documents are comprised largely of natural language but with some structuring conventions. These are denoted as PSLNL (partially-structured, largely-natural-language) documents (Ma, Shepherd & Zhang 2002). For example, most job advertisements are laid out as a sequence of informative regions including job description, requirements, salary, deadline, etc. In other words, they often contain similar semantic and orthographic structures. Making use of these particularities, we propose a new method to reduce the size of the feature set for PSLNL documents without compromising the classification accuracy. First, document synopses are extracted, which contain the most informative data; then succinct feature vectors are constructed based on these synopses. Finally, a decision tree machine learning algorithm uses these feature vectors to classify the documents.

Note that the method in this paper is described in the context of PSLNL documents. In fact, the requirement here is the ability to partition documents into regions. In PSLNL documents, this is achieved via machine learning on a set of basic orthographic features. We postulate that the method would apply equally to semistructured documents, such as Web pages, where markup features are available to assist in the regioning process.

In this paper we present a method for document classification that exploits information extraction techniques to derive small feature vectors that discriminate well between document classes. Section 2 studies previous approaches to feature selection. A description of a system incorporating the proposed

method is presented in Section 3. Section 4 covers the synopsis generation and text classification processes in detail. After giving some preliminary experimental results on PSLNL documents in Section 5, conclusions and future are discussed in Section 6.

2 Related Work

Recently, a growing number of statistical and machine learning methods have been applied to solve text classification problems. The major difficulty in doing this is the high-dimensionality of conventional document feature spaces. Words, or, more rarely, phrases, are typically used as features. Even with moderated-size document collections, the size of the feature space can reach hundreds of thousands. Most existing machine learning algorithms are not designed to deal with such large feature spaces, and applying them naively is computationally infeasible. In addition, the use of words as features can cause other problems: the classification might be misled by general words and discriminating words might not be used effectively. It is therefore highly desirable to reduce the size of the feature space.

Enhanced feature extraction methods either remove non-informative terms (feature selection) or combine and transform original terms to form new features (re-parameterisation).

The simplest feature selection technique, *Document Frequency Thresholding*, discards words whose document frequency is not within some predetermined range. This technique is based on the assumption that neither rare words nor common words are useful as classification discriminators. Applying this cutoff can reduce the size of feature matrices but, as a consequence, may miss out important terms. (Lewis & Ringuette 1994) use the *Information Gain* measure to rank and select the most category-predictive words. This heuristic, which was first introduced in (Quinlan 1986), is based on the probability of a document containing a given term and belonging to a given category. According to (Y.Yang & J.O.Pedersen 1997), this method has the drawback of being strongly affected by the marginal probabilities of terms, i.e. rare terms will be given a higher score. Another method, from (Schutze, Hull & Pedersen 1995), measures the goodness of a term as a function of its χ^2 statistic, which indicates the lack of independence between a word and a category. The χ^2 statistic, however, is known to be unreliable for low-frequency terms (Dunning 1993). (Y.Yang & J.Wilbur 1996) studied an aggressive word removal strategy using document clustering techniques. Probabilistic *Word Strength* is computed based on word distribution over related texts, which rates how informative a word is. A domain specific stoplist can then be constructed that contains words to be removed from the text documents. This technique was shown to rapidly lose accuracy once more than 50% of the words were removed (Y.Yang & J.O.Pedersen 1997).

The re-parameterisation branch of automatic feature extraction aims to reduce the size of feature vectors by constructing new features as combinations or transformations of lower level features. *Latent Semantic Indexing* (LSI) is one such method (Deerwester, Dumais, Furnas, Landauer & Harshman 1990, E.Wiener, J.O.Pedersen & A.S.Weigend 1995). It assumes there is an underlying ("latent") structure in word usage. This structure is estimated using singular-value decomposition technique. Classification is then performed using the database of these singular values (M.W.Berry, S.T.Dumais & G.W.O'brien 1995).

All existing feature extraction methods first construct a feature space as the set of all non-trivial terms[1] that appear in the documents. After that, different heuristics are used to refine the original feature set and reduce its size. Our approach differs from these in attempting to extract small, yet effectively discriminating vectors, in the first instance. To do this, it exploits the fact that documents in particular categories tend to exhibit characteristic stylistic structures.

3 System Overview

Our approach to reducing the dimensionality of the document feature space is to extract a small subset of the information in the document which effectively represents the contents of the document. In (Ma et al. 2002), we described a system that can effectively partition a PSLNL document into regions. Each region is assumed to contain information of a particular kind, often structured as a sequence of records. The system also extracts a title for each region and a heading for the whole document. We use this system for the present application, except that we ignore the region contents, which contain large numbers of words, and use just the document heading and the region titles to provide a concise summary of the document. The intuition here is that the document heading and region titles contain precisely the keywords that best characterise the document class. In addition, the presence and absence of specific titles can also provide an indication of the document's class.

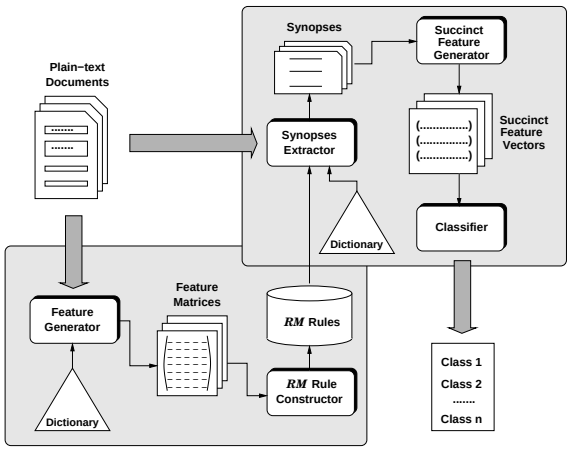

Figure 1: *System Architecture*

Our text classification system (Figure 1) consists of five components:

- The Feature Generator uses orthographic and semantic knowledge to produce a set of feature vectors (*feature matrix*) for each document.
- The RM Rule Constructor takes feature matrices for a collection of training documents, and employs the C4.5 decision tree algorithm (Quinlan 1993) to construct a set of rules for partitioning documents into regions and extracting a title for each region. These rules are denoted *Region-title* Mapping (RM) Rules.
- The Synopses Extractor uses the *RM Rules* to extract the heading and region titles from input

[1]The definition of trivial terms (e.g. stop words) may vary and is application-specific. In general, keeping non-trivial terms and removing trivial ones do not help much in reducing the size of feature space.

Name of Feature	Description of Feature	Values
Orthographic features		
isUpperCaseLine	all alphabetics are upper case	0, 1
isFirstUpperCaseLine	each word starts with upper case	0, 1
dateLine	contains date information	0, 1
webLine	contains a URL	0, 1
endWithColon	ends with colon	0, 1
startsWithDigit	starts with number	0, 1
startsWithChar	starts with non-alphanumeric	0, 1
unPunctuatedLine	not normal punctuated	0, 1
Relative features		
lineNumber	line number in document body	numeric
preLineIsBlank	after a blank line	0, 1
nextLineIsBlank	before a blank line	0, 1
nearBreak	neighbour is a break line	0, 1
isShortLine	shorter than $\frac{max(lineLength)}{2}$	0, 1
sameAsPreLine	same format as previous line	0, 1
sameAsNextLine	same format as next line	0, 1
Semantic feature		
keywordFeatured	0 = contains no keyword 1 = contains a keyword, but no other features from above 2 = contains a keyword, and some other features from above	0, 1, 2
Region classification		
class	0 = normal region 1 = region title 2 = heading region	0, 1, 2

Figure 2: Feature vectors

documents. The information from the heading and region titles forms a *Synopsis* of the document.

- The Succinct Feature Generator uses the *Synopses* to produce *Succinct Feature Vectors* (SFVs) The features comprise the set of possible words from the headings and region titles. The values associated with these features are binary, indicating the appearance, or not, of a particular word in a particular document.
- The Classifier uses SFVs as a basis for classifying input documents. The Classifier is produced by applying the C4.5 algorithm to the SFVs of the training documents.

The system requires two distinct training phases: (1) training to recognise regions[2] and region titles, and (2) training to classify documents based on succinct feature vectors. Once training is complete, the system can be deployed for document classification. There are two major phases in the classification process: (1) extracting the synopses via the RM rules, and (2) classifying documents using the extracted synopses.

4 Classification via Synopses

4.1 Region Decomposition

For documents of a particular type (e.g. calls-for-papers), there is a finite set of pieces of information that might potentially be included. For example, most calls-for-papers would contain the title of the conference, where it is being held, dates for paper submission, topics-of-interest, details about the program committee members, etc. Specific kinds of information (e.g. important dates) are typically located together in a distinct area of the document, generally prefixed by a title. We denote such areas as *regions* and denote the fact that they contain a specific kind of information as *semantic coherence*. Any given document will contain a subset of the possible regions for its document type.

4.2 Feature Vector Generation

The first step in region identification is to produce a set of feature vectors for the document. We currently produce one feature vector for each line; however, this is not forced by our method and could be modified for other kinds of documents. The kinds of features that we consider are: orthographic, relative and semantic. The choice of specific orthographic and relative features was based on observation of typical conventions in PSLNL documents, and refined by experimentation. Our experiments showed that, while the choice of features can have a substantial impact on performance, the features in Table 2 are suitable for many classes of PSLNL documents. For other kinds of documents, different kinds of features would need to be chosen.

Orthographic features are style characteristics that are used to convey the role of words or sentences. Examples include using all-uppercase letters for headings, first-uppercase letters for proper names, starting text in a particular column, a star (bullet) at the beginning of a line, etc. The primary assumption used in feature generation is that authors want readers to notice the most important information easily, and so they typically mark the information by a distinguishing layout. Also, for the purpose of coherence, authors generally use the same layout to express related pieces of information.

Relative features indicate the position of a line within the document and its relationship to neighbouring lines. These kinds of features are important

[2]Proteus (Embley, Campbell, Jiang, Ng, Smith, Liddle & Quass 1998)(Embley, Jiang & Ng 1999) works on HTML documents where record boundries are mostly discovered by processing tags. On the contrary, there is no artificial tags inside PSLNL documents. Inferring relation from hypertext is a special case of that from any plain-text document. In fact, tagged documents (eg. HTMLs) provide more parsing clues than PSLNL documents do since the tags can be considered as syntactic features. As a result if a system works on PSLNL documents, the system can be adapted to deal with tagged documents easily.

```
Is_Head_Section_Rules(Vector feature): if
{
    (lineNumber<=pos1, !endWithColn);
    (lineNumber<=pos2, !isFirstUpperCaseLine, !keywordFeatured);
    (lineNumber<=pos3, isUpperCaseLine, !keywordFeatured);
    (lineNumber<=pos4, dateLine);
    (lineNumber<=pos5, webLine);
    (lineNumber<=pos6, isFirstUpperCaseLine, unPunctuatedLine);
    ... ...
}
Is_Region_Title_Rules(Vector feature): if
{
    (keywordFeatured, orthographic features);
    (!webLine, preLineIsBlank, isShortLine, (relative and orthographic features));
    (endWithColon);
    (startsWithDigit, !endsWithColon, previousLineIsBlank, nextLineIsBlank);
    (unPunctuatedLine, keyworkFeatured(weak), !dateLine, isShortLine, preLineIsBlank);
}
Is_Region_Rules(String line): if
{
    In_between_two_title();
}
```

Figure 3: Region extraction rules

because the significance of orthographic features is often affected by their context. For example, a line that is preceded by a blank line and followed by a line of hyphens is significant (most likely a heading), almost regardless of its own orthographic features. Similarly, lines that appear near the top of the document typically have added importance. Another example might be two successive lines that have precisely the same format (e.g. both start with an asterisk) indicating a list.

Semantic features are domain-specific keywords that help to identify particular semantic contexts within the document. These contexts are used to assist in identifying what kind of information is expected in a given region.

Note that orthographic and relative features are dependent on the document format (e.g. plain text documents have different features to HTML document) but are largely independent of any domain. Semantic features, on the other hand, are domain-specific and need to be reconsidered for each new application of our method. It also worth noting that preliminary experiments established that these different kinds of features need to be considered in combination in order to achieve satisfactory classification results.

With the help of experiments, we identified fifteen common features and one domain-specific feature as shown as in Table 2. Most of the features are self-explanatory. The *keywordFeatured* feature indicates whether a line contains one of the domain-specific keywords supplied by the user, and also indicates whether the line contains any other orthographic features. The *class* "feature" indicates the correct classification for the region in the training data; it will be output from the classifier after it is constructed.

4.3 Rule Construction

In the rule construction phase, the system builds a system that can recognise important features in documents. This is phrased as a classification task, where we aim to assign each line of the file into one of three possible classes:

- *heading* - introductory heading typically giving name, affiliation, location, etc.
- *region* - semantically coherent blocks of text, such as a list of topics
- *region title* - heading associated with a particular region, typically giving region type

The application is complicated by the presence of dependencies among concepts and can be cast as a multiple predicate learning problem.

As explained in section 3, regioning is the first step in parsing the whole document. In the extraction rule constructor, we use the C4.5 (Quinlan 1993) decision-tree induction algorithm to generate a region classifier. C4.5 is a system that learns decision-tree classifiers from a set of pre-classified cases, each described by a vector of attribute (feature) values and the class for that region. In our application, all attribute values are integers, with meanings as indicated in Table 2.

Each line in a document is associated with a feature-vector (a sequence of attribute values). Each feature is used as an attribute of the learning system. The document can be viewed as a feature matrix. The aim of the learning system is to classify each line in the document. For the purpose of generating succinct feature vectors, we discard all lines belonging to regions, and retain only those from the heading and region titles.

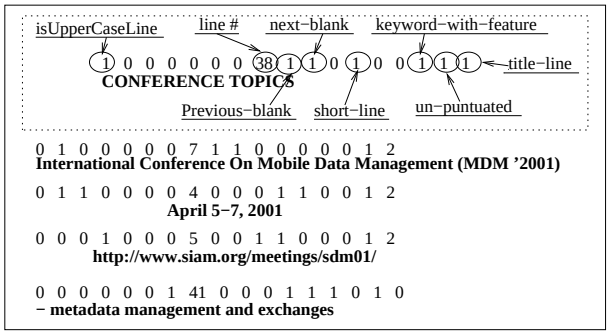

Figure 4: **Example of feature vectors and classes**

Figure 4 shows three example of vectors associated with different line classes. Line 7 of the document, with features of isFirstUpperCaseLine, previousLineIsBlank, nextLineIsBlank and unPunctuatedLine, belongs to the *heading* class. Line 38, with features of isUpperCaseLine, preLineIsBlank, nextLineIsBlank, isShortLine, unPunctuatedLine and keywordFeatured, belongs to the *title* class. Line 41, with features of startsWithChar, isShortLine, sameAsPreLine, sameAsNextLine and unPunctuatedLine, falls in the *region* class.

After applying the C4.5 decision-tree classifier, a set of classification rules is generated which attempts

Input: plain-text document $d = \bigcup_{k=1}^{n} \{l_k\}(l_k = line_k)$, *RM Rules*, *dictionary* $\mathcal{T}$

Output: heading h and *titles* $\mathcal{T} = \bigcup_{i=1} \{f_i\}$ of other *regions*

begin:

Step 1: /*feature vectors generation*/
for $k = 1, \ldots, n$ **do** fv_k:=feature vector of l_k;

Step 2: /*heading extraction*/
h:=$\emptyset$;
j:=2;
while($is\_Head\_Section\_Rules(fv_{j-1})||is\_Head\_Section\_Rules(fv_j)$)
h:=$h \bigcup l_j$, j:=j+1;

Step 3: /*coarse region title identification*/
c:=0
for $m = j, \ldots, n$
if ($Is\_Region\_Title\_Rules(fv_m)$) **then** c:=c+1, $t_c := l_m$;
else $r_c := r_c \bigcup l_m$;

Step 4: /*fine region title identification*/
for $p = 1, \ldots, c$ **do** $f_p := classifyGroup(t_i, \mathcal{T})$;
for $p = 1, \ldots, c-1$ **do**
if ($f_p = f_{p+1}$) **then** $r_i := r_p \bigcup r_{r+1}$, $t_{p+1} := \emptyset$, $f_{p+1} := \emptyset$, $r_{p+1} := \emptyset$;
$\mathcal{T} := \emptyset$
for all $t_i \neq \emptyset$ **do** $\mathcal{T} := \mathcal{T} \bigcup \{t_i\}$;

end

Figure 5: Algorithm for region extraction

to identify which class a particular line belongs to. Examples of these rules are shown in Table 3. Each decision rule emphasises one particular feature. For example, lineNumber is most important when class *heading* is classified; orthographic and semantic features are considered more important in determining class *title*.

4.4 Region Decomposition

4.4.1 Head Recognition

Information belonging to the *heading* region appears at the top of the document and has special orthographic features. For example, a conference CFPs contains at least the conference title, possibly with additional information such as URLs, date and location, etc. The $Is\_Head\_Section\_Rule$ considers primarily orthographic and relative features to identify the lines in the *heading* region.

Other information might also appear near the top of the document, such as reminders and notices which indicate the type and purpose of the document. This information has no relevance to the heading section, and is not part of the information that we wish to extract from the document.

The information from the heading region is kept as a set h for use in subsequent SFV generation. Determining where the heading region ends could not be done reliably with the features we used, and an additional heuristic was used to ensure that we did not stop heading region collection prematurely (see step 2 of the algorithm in Figure 5). The collection process uses not just the line classifier, but also takes account of the relationship between adjacent lines.

4.4.2 Region Recognition

The remainder of the document after the heading region contains most of the detailed information to be extracted from the document. However, parsing this section is also difficult because of its loose structure. Our approach is to use different granularities of parsing. Coarse parsing makes a preliminary identification of document regions using primarily orthographic and relative features. Fine parsing then refines the regions using a combination of orthographic and semantic features. The goal is to produce labelled blocks of text, where the labels (region titles) can be fed into the SFV generation phase. The complete algorithm for region decomposition is given as Figure 5.

Step 3 (in Figure 5) corresponds to coarse parsing which tries to identify possible regions (r_is) and region titles (t_is). However, this process is approximate and so the system may have failed to recognise some title lines as titles or may have misclassified other non-title lines as titles. Step 4 (in Figure 5) corresponds to fine parsing, which tries to correct any misclassification from the coarse parsing, and thus reliably identify region titles.

4.5 Text Classification

The next stage of the classification process takes the $\mathcal{T}$ and h values produced by the region extractor and combines them to produce a *synopsis* of the document. This is further processed into a keyword-based feature vector that represents the document and is used as the basis for classification (since these feature vectors are relatively small compared to the ones typically used in document classification, we denote them *succinct feature vectors*). The classifier itself is produced via machine learning on the succinct feature vectors of a pre-classified training set.

The synopsis is produced by forming a set of all words from $\mathcal{T}$ and h. This set is then refined via the standard text retrieval stemming and stopword-removal operations. Finally, various classes of words that do not aid in classifying the type of the document are replaced by their class name. For example, the location of a conference might be useful in finding the program for a specific conference, but it is not a useful feature in determining whether a given document is a conference program or not. We thus replace proper names by the class to which they belong (e.g. country, person, organisation). Similarly, all dates, times and URLs are replaced by their class names (e.g. `www.acm.org` and `www.cse.unsw.edu.au` would both be replaced by the name URL). The effect of all of these transformations is that we have a considerably smaller set of words to deal with, but without losing substantial discriminatory power for the task of classification.

We also perform one transformation that actually increases the size of the words: we distinguish be-

tween the occurrence of a term in h or $\mathcal{T}$. The rationale for doing this is that terms appearing in h typically have a different function to the same terms appearing in $\mathcal{T}$. Note that this transformation actually takes place *before* the transformations described above.

Given a training corpus, we form a feature space by forming a union of all of the individual synopses. The individual terms form the features (dimensions) of this space. A succinct feature vector can be generated for a given synopsis by assigning 1 for each feature that appears in that synopsis, and 0 for any term that does not appear. This will generally lead to quite "sparse" feature vectors. In order to generate a classifier for this training set, the set of succinct feature vector and the classification for each document are given to a decision tree learning algorithm. The result is a classifier that can take new documents and accurately map them to an appropriate class, where each incoming document needs to processed as described above to produce a succinct feature vector.

5 Experimental Results

We have developed a system to implement the method proposed above. The system was based on the region extractor implemented in some previous work (Ma et al. 2002) on content extraction, extended in a straightforward way to generate succinct feature vectors. In this section, we report some preliminary experimental results on its effectiveness.

The aim of our experiments was to provide some preliminary evidence on how effective the new method was and also give some idea of its robustness. In particular, we aimed to measure the overall accuracy of the method and see how the accuracy varied with the size of the training set. The *accuracy* measure used in this work is the proportion of documents correctly classified against all documents classified.

Experiments were carried out on a data set of 4400 email documents posted to the DBWorld mailing list during 2000 and 2001. This data set was manually classified into nine categories (Figure 6). For each experiment, the data was partitioned into two disjoint sets (some documents formed the training set Tr[3], the rest formed the testing set Te). The classifier was trained using Tr and then all of the documents in Te were classified using this classifier and the accuracy measured. The size of training sets ranged from 200 to 4000. For each training set size, the accuracy experiment was repeated for ten randomly-generated combinations of (Tr, Te). The accuracy results reported in Figure 7 are a simple mean of the accuracy over the ten trials for each training set size.

Category	#documents
Books	199
BookCfch	4
Conference	2626
Grants	11
Jobs	867
JournalCfp	131
Journals	231
News	244
Software	87
Total	4400

Figure 6: *The Data Set*

[3] Tr, which is chosen from each category with same proportion, contains documents of all categories.

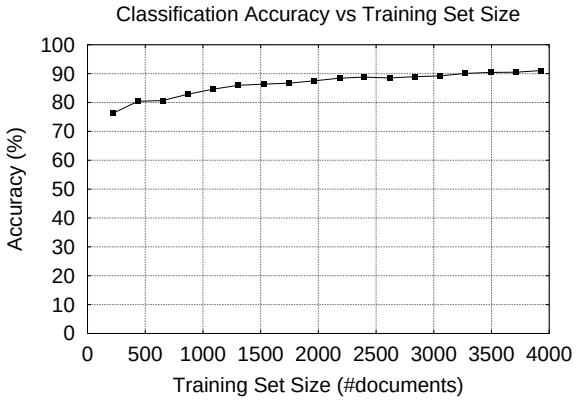

Figure 7: *Experimental Results*

The results indicate an average accuracy of around 85%, with accuracies above 75% even for quite small training sets. This compares favourably with previous systems for classifying PSLNL documents, whose accuracy is typically in the range 80%-90% (e.g. (Y.Yang & J.Wilbur 1996), SCAR in (I.Moulinier, G.Raskinis & Ganascia 1996), (A.McCallum, R.Rosenfeld, T.Mitchell & A.Y.Ng 1998), (E.Wiener et al. 1995), etc.). It is important to note that the average number of "features" for our classifier was around 2200 (cf. the size of around 24000 if vector space model was used together with stemming and stopword removal techniques). This is *considerably* smaller than the feature spaces for the other methods.

6 Conclusion

In this paper, we describe a novel method for extracting features for document classification, based on the notion of identifying regions in the document and then using the titles of these regions to derive a relatively small feature space. The accuracy of the classifiers produced from this feature space is comparable to those reported in previous document classification efforts using much larger feature spaces.

While these results are impressive, there remain further possibilities for refining the approach. For example, it may be possible to use semantic similarities of terms to further normalise the term set from the titles and headings, thus reducing the size of succinct feature vectors even further, without reducing the classification effectiveness.

References

A.McCallum, R.Rosenfeld, T.Mitchell & A.Y.Ng (1998), Improving text classification by shrinkage in a hierarchy of classes, *in* 'Inter'l Conf. on Machine Learning'.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. & Harshman, R. (1990), 'Indexing by latent semantic analysis', *Journal of the American Society for Information Science* .

Dunning, T. (1993), Accurate methods for the statistics of surprise and coincidence, *in* 'Computational Linguistics'.

Embley, D. W., Campbell, D. M., Jiang, Y. S., Ng, Y.-K., Smith, R. D., Liddle, S. W. & Quass, D. W. (1998), A conceptual-modeling approach

to extracting data from the web, *in* 'ER'98, Proceedings of International Conference on Conceptual Modeling'.

Embley, D. W., Jiang, Y. S. & Ng, Y.-K. (1999), Record-boundary discovery in web documents, *in* 'SIGMOD'.

E.Wiener, J.O.Pedersen & A.S.Weigend (1995), A neural network approach to topic spotting, *in* 'Symposium on Document Analysis and Information Retrieval'.

I.Moulinier, G.Raskinis & Ganascia, J.-G. (1996), Text categorization: a symbolic approach, *in* 'Symposium on Document Analysis and Information Retrieval'.

K.Aas & L.Eikvil (1999), Text categorisation: A survey, Technical report, Norweigan Computing Center.

Lewis, D. & Ringuette, M. (1994), 'A comparison of two learning algorithms for text categorization', *Symposium on Document Analysis and Information Retrieval* .

Ma, L., Shepherd, J. & Zhang, Y. (2002), Extracting information from semistructured data, *in* 'International Conference on Web-Age Information Management'.

M.W.Berry, S.T.Dumais & G.W.O'brien (1995), Using linear algebra for intelligent information retrieval, Technical report, The Society for Industrial and Applied Mathematics(SIAM) Review.

Quinlan, J. R. (1986), Induction of decision trees, *in* 'Machine Learning'.

Quinlan, J. R. (1993), 'C4.5: Programs for machine learning'.

Schutze, H., Hull, D. A. & Pedersen, J. O. (1995), A comparison of classifiers and document representations for the routing problem, *in* 'Research and Development in Information Retrieval'.

Y.Yang & J.O.Pedersen (1997), A comparative study on feature selection in text categorization, *in* 'International Conference on Machine Learning'.

Y.Yang & J.Wilbur (1996), 'Using corpus statistics to remove redundant words', *JASIS, Journal of the American Society for Information Science and Technology* .