Two-Level Text Classification Using Hybrid Machine Learning Techniques

Nandita Tripathi

A thesis submitted in partial fulfilment of the
requirements of the University of Sunderland
for the degree of Doctor of Philosophy

July 2012

School of Computing, Engineering and Technology
University of Sunderland
Sunderland
United Kingdom

Abstract

Nowadays, documents are increasingly being associated with multi-level category hierarchies rather than a flat category scheme. To access these documents in real time, we need fast automatic methods to navigate these hierarchies. Today's vast data repositories such as the web also contain many broad domains of data which are quite distinct from each other e.g. medicine, education, sports and politics. Each domain constitutes a subspace of the data within which the documents are similar to each other but quite distinct from the documents in another subspace. The data within these domains is frequently further divided into many subcategories.

Subspace Learning is a technique popular with non-text domains such as image recognition to increase speed and accuracy. Subspace analysis lends itself naturally to the idea of hybrid classifiers. Each subspace can be processed by a classifier best suited to the characteristics of that particular subspace. Instead of using the complete set of full space feature dimensions, classifier performances can be boosted by using only a subset of the dimensions.

This thesis presents a novel hybrid parallel architecture using separate classifiers trained on separate subspaces to improve two-level text classification. The classifier to be used on a particular input and the relevant feature subset to be extracted is determined dynamically by using a novel method based on the maximum significance value. A novel vector representation which enhances the distinction between classes within the subspace is also developed. This novel system, the Hybrid Parallel Classifier, was compared against the baselines of several single classifiers such as the Multilayer Perceptron and was found to be faster and have higher two-level classification accuracies. The improvement in performance achieved was even higher when dealing with more complex category hierarchies.

Acknowledgement

I would like to thank Dr. Michael Oakes, my Director of Studies without whose encouragement, critical review and constant support, this research would not have reached its completion. His expertise in the field of text classification helped to position this work in the right context.

I would also like to thank Professor Stefan Wermter, my initial Director of Studies and subsequently external co-supervisor for his continued interest and guidance throughout the period of my research.

Thanks also go to Dr. Albert Bokma and Professor Fawaz Al-Anzi for all the help and encouragement extended to me during the course of my work.

I would also like to extend my thanks to Professor Eric Gaussier of Université Joseph Fourier (Grenoble I), France for permitting the use of their compiled LSHTC corpus and to the Weka mailing list administrator, Dr. Mark Hall, for promptly replying to all my queries on Weka.

Finally, I would like thank my husband for his constant moral support and my two kids for letting me take away time that was rightfully theirs.

This work is dedicated to the memory of my father without whose unconditional support and belief in my abilities, I would not be where I am today

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background & Motivation

Documents today are often maintained in category hierarchies rather than a flat classification system. As the volume and diversity of documents grow, so do the size and complexity of the corresponding category hierarchies. Documents collected for a specific purpose such as collections of medical documents (MEDLINE), patent documents (WIPO) and news articles (RCV1) are all structured in a hierarchy. The Reuters Corpus (RCV1) has news articles classified in a hierarchy of up to five levels. For example, a Reuters news item has tags MCAT(Markets)/ M14(Commodity Markets)/ M141(Soft Commodities) to denote the three levels of categories associated with it. On the web, Yahoo! and DMOZ are two examples of systems which follow a structured document catalogue. An exhaustive study conducted in 2004 found the Yahoo! directory at that time to contain 292,216 categories in a 16-level hierarchy. Traditional classifiers cannot take advantage of this hierarchical information. The hierarchy has to be flattened to a single level, either manually or through a program, for the application of these classifiers. Flattening results in a huge number of categories which have to be differentiated by a single classifier. This greatly degrades the performance of many classifiers. Furthermore, the information inherent in the hierarchy is lost during flattening and a single classifier is not able to focus on differences between categories at the lower levels of a hierarchy. To be able to access hierarchically-classified documents in real time, we need fast automatic methods to navigate these hierarchies. As data gets progressively sparser at deeper levels of a hierarchy, it is more appropriate to concentrate on two or three topic levels for classification. Instead of using a classification algorithm at the first level, methods which can directly point to a relevant main topic should be explored. Such methods will help in increasing the search/classification speeds.

Subspace learning is an area popular with non-text domains such as image recognition to increase speed and accuracy. The vast data space in today's world is divided into many subspaces which are quite different from each other, e.g. medicine and politics. Since each subspace can be viewed as an independent dataset, separate classifiers can be used to process separate subspaces. Instead of using the complete set of full space feature dimensions, classifier performances can be boosted by using only a subset of the dimensions. In this work we explore hybrid classifiers based on semantic data subspaces as a means to improve two-level classification of text documents.

1.2 Research Aims and Objectives

1.2.1 The Aim

The aim of our research is to improve the speed and accuracy of automatic document classification in the presence of category hierarchies. This will be done by developing novel hybrid machine learning techniques and novel vector representations based on the semantic content of news and text documents.

1.2.2 The Objectives

The objectives include the following research and experimental tasks:

1. Conduct a literature review on the current state of multilevel text classification systems using machine learning methods with emphasis on hierarchical classification and subspace learning.
2. Propose a new vector representation suitable for two-level learning.
3. Conduct a literature review on the currently available methods of classifier combination.
4. Research various classifier combination methods to improve two-level text classification.
5. Propose a new hybrid architecture for improved two-level learning using the new proposed vector representation.
6. Evaluate the performance of the new proposed hybrid architecture using various performance methods.

1.3 Research Questions and Hypothesis

1.3.1 Research Hypothesis

- The use of separate classifiers for separate subspaces will improve overall subspace classification accuracy and learning time and lead to improved two-level classification of text documents.

1.3.2 Research Questions

Research Question 1) Is it possible to devise a method to quickly direct the document search to a relevant document subspace by examining only a single input query vector?

Research Question 2) Can we develop a classification method which directs the classification from all possible classes to a relevant subspace of classes?

Research Question 3) Is it possible to have a vector representation that focuses on the relative importance of keywords within a data subspace?

1.4 Novelty / Original Contributions of the Thesis

- *Subspace Detection using Maximum Significance Value:* We developed a method of detecting the subspace (level 1 category) of a document from the document vector itself. The document was represented using significance vectors. The components of a significance vector are the categories present in the data. Since we considered a two level classification system, categories from both level 1 and level 2 were represented in the significance vector. Each entry in a document significance vector gives the significance of the document to that particular category. We proposed that the maximum numerical value entry among the level 1 categories indicated the level 1 category or the subspace that the document was most likely to belong to. We called this the *Maximum Significance Value*. The relevant subspace of a new test document is thus detected from the document vector itself using the maximum significance value. This value can be calculated in O(k) time where k is the number of level 1 topics. Hence this is a very fast method of subspace detection.

- *Conditional Significance Vector Representation:* We developed this new vector representation which is based on term frequencies *within* a given subspace. This enhances the distinction between subcategories within a subspace when compared to a normal significance vector which uses term frequencies across the full data space.
- *Hybrid Parallel Classifier Architecture:* We developed and tested this new architecture which takes advantage of the semantic subspaces present in the data and improves classification at subcategory level
- *Parallel Classifier Architecture:* We also tested this special case of our hybrid parallel classifier architecture using the *same type* of classifier for all subspaces. This significantly reduces classifier training and test timings along with improving classification at subcategory level.
- *Subspace Based Dimensionality Reduction:* We use only the vector components relevant to a document's subspace in the hybrid classifiers and show that such dimensionality reduction improves subspace learning.

We used the Reuters Headlines as well as Reuters Full Text (Headlines + Body Text) for our experiments showing that Reuters Headlines are better at classifying news items than Reuters Full Text. We also used a corpus (LSHTC) drawn from real web data showing that our methods are equally applicable to the web. Two different baselines, significance vectors and tf-idf, were used to provide a comparative evaluation of our proposed system. The performance metrics used for the comparisons were classification accuracy and training/test timings. We also ran statistical significance tests on these values. The results were shown to be statistically significant in all cases. We have applied our novel method to the field of text classification. Our thesis also discusses how these techniques can be applied to other domains.

1.5 Structure of the Thesis

This thesis is divided into seven chapters including the current introduction chapter.

Chapter 2 looks at the current state of text classification where the data is arranged in the form of multilevel hierarchies. It also presents subspace learning, a concept popular in pattern recognition to deal with high dimensions and discusses its application to text classification. Classifier combinations are also explored as a means to improve classification accuracies.

Chapter 3 introduces the novel techniques of Maximum Significance Value, Conditional Significance Vectors and Hybrid Parallel Classifiers along with the experimental methodology. The explanation is divided into two phases representing the sequence of development of the novel techniques. The two test corpora, the Reuters Corpus (RCV1) and the Large Scale Hierarchical Text Classification (LSHTC) dataset are also discussed in detail in this chapter.

Chapter 4 presents the results of Conditional Significance Vectors (Phase I). These results are compared with two different baselines vector formats – the tf-idf vector and the standard Significance Vector. Experiments are conducted on two datasets, Reuters Headlines and Reuters Full Text, extracted from the Reuters Corpus and also on the LSHTC dataset using two levels of topic hierarchy.

Chapter 5 presents the results of Phase II of the experiments conducted on the final novel architecture – the Hybrid Parallel Classifier. Here again, the results are compared with two different baselines, full data classification using tf-idf and significance vectors. Two sets of experiments are presented here. Experiment Set A presents the hybrid combination of the Multilayer Perceptron (MLP) with a number of other classifiers while the Experiment Set B presents the hybrid combination of a wide variety of classifiers.

Chapter 6 presents the results of a special case of the Hybrid Parallel Classifier – the *Parallel Classifier* which combines different classifiers of the same type in a parallel combination. These results are again compared with the baselines using tf-idf and significance vectors. This chapter presents the effect on classifier timings (both training and test) as well as classification accuracy.

Chapter 7 concludes the thesis with a summary of the work undertaken. It discusses the outcomes of this research and its implication for the fields of text classification and classifier theory. It also discusses the applicability of this work to other domains and suggests the scope for future work in this area.

Chapter 2

Multi-Level Text Classification

2.1 Machine Learning in Multi-Level Text Classification

Traditional methods of automatic text classification deal with a number of categories on a single level i.e. they implement a flat classification scheme. Category hierarchies, however, are a convenient way of arranging huge amounts of data in a manageable form. A large number of organisations are nowadays associating their data with multilevel category schemes. In this chapter we look at the traditional methods of text classification, the presence of multi-level data in today's world, and various methods of extending traditional classifiers to deal with multi-level data.

2.1.1 Text classification with single level data

Machine Learning techniques for text classification have been applied to document classification (Li & Jain, 1998), news classification (Bacan, Pandzic, & Gulija, 2005), spam filtering (Androutsopoulos, Koutsias, Chandrinos, Paliouras, & Spyropoulos, 2000), automatic essay grading (Larkey L. , 1998) and sentiment analysis (Pang & Lee, 2008) to name a few.

Comparative studies of text classifiers have been carried out by various researchers.

Sebastiani (2002) compared 42 results on text classification published by various researchers and concluded that the best text classifiers were Support Vector Machines (SVM), regression based classifiers, example based classifiers and boosting methods. These were closely followed by Neural Networks and on-line classifiers. The worst performing classifiers were found to be the Naïve Bayes and Rocchio classifiers. He stated that it was difficult to draw any conclusions about decision trees though one of the results (Dumais et al) showed the performance of a decision tree to be very near that of the SVM.

Yang (1999) compared some previously published results along with her new results on five different versions of the Reuters Corpus. She concluded that kNN was one of the best performing classifiers followed by tree-based and rule-based classifiers. Naïve Bayes and Rocchio were determined to have poor performances. Joachims (Joachims, 1998) compared the performance of several SVMs based on polynomial and RBF kernels against that of four traditional baselines – Naïve Bayes, kNN, Rocchio and the C4.5 decision tree. He reported that all SVMs performed better than all the baselines with kNN being the best performer among the baselines. A subsequent paper by Yang & Liu (1999) disputed this finding of Joachims and compared five learning methods – Support Vector Machines (SVM), Neural Networks (NNet), LLSF (a method developed by Yang), Naïve Bayes (NB) and their own version of k-nearest neighbours (kNN). They concluded that SVM and kNN were the best text classifiers while NB was the worst.

Dumais et al. (1998) compared five methods - Naïve Bayes, BayesNet, Decision trees, SVM and a variation of Rocchio's algorithm on Reuters 21578. They concluded that SVM was the best text classifier with trees coming in second and Bayesian classifiers last. Among Bayesian classifiers, BayesNet performed better than Naïve Bayes. Basu et al. (2003) compared SVM with Neural Networks on the Reuters 21578 collection and concluded that the performance of SVM was significantly better than that of Neural Networks. Lewis et al. (2004) compared SVM, weighted kNN and Rocchio on the new Reuters RCV1 corpus using both micro and macro averaged effectiveness measures. They found that SVM performed the best followed by weighted kNN. Rocchio's performance was below both SVM and kNN.

Hence the general consensus in research seems to be that SVM is the best text classifier while Naïve Bayes is among the worst. However, Giorgetti & Sebastiani (2003) compared SVM and Naïve Bayes (the best and the worst text performers) with a dictionary based approach in a multiclass setting for automated survey coding. They reported that although both the methods outperformed the dictionary method, Naïve Bayes outperformed SVM by a small margin. This calls into question the capabilities of SVM in a multiclass setting. SVMs have originally been developed for binary classification and their applications to multiclass classification needs to be explored further. Meanwhile, interest in Naïve Bayes still continues unabated. Kim et al. (2006) suggested a Poisson Naive Bayes implementation with weight-enhancing method and tested it on the Reuters 21578 and 20 Newsgroups text corpora. They reported a performance approaching that of the SVM on the same corpora.

Text classification is intuitively a multiclass problem. However a lot of earlier work using SVMs has been done on binary text classification. SVMs are the best performers in binary classification but their logic of a separating hyperplane cannot be directly applied to a multiclass setting (Li, Zhu, & Ogihara, 2003). Multiclass problems are usually solved by decomposing them into a set of binary problems. Two approaches used for this are one-vs-rest and one-vs-one. The one-vs-rest approach has *n* binary classifiers to decide the classification for each of the *n* categories while the one-vs-one approach had ${}^{n}C_{2}$ or n(n-1)/2 binary classifiers. The one-vs-one is the most exhaustive approach comparing each category with every other category. The outcomes of these classifiers are then combined to give the final result. Both these approaches have a significant bearing on the classifier training times. The training time for a binary SVM is $O(n^{k})$ where 1.7<=k<=2.1 (Li, Zhu, & Ogihara, 2003) where n is the number of training instances. Multiclass classification using SVM would require training many SVMs resulting in a high overall classifier training time.

To bypass multiple binary classifications, Li et al. (2003) propose an algorithm called GDA based on Generalised Singular Value Decomposition (GSVD) for direct multiclass classification. They compare their GDA algorithm with Naïve Bayes, kNN and one-vs-rest SVM on a number of datasets including Reuters top10 which is a subset of Reuters 21578 consisting of the ten most frequently occurring categories. Naïve Bayes and kNN can directly handle multiclass classification and do not need multiple binary classifiers for this task. They report that Naïve Bayes was the best performing classifier on Reuters top10.

A recent report by Henderson (2009) compares four algorithms, Naïve Bayes, SVM, kNN and multiclass Rocchio (also called Centroid) for multiclass classification on a subset of data from the open directory project (ODP/DMOZ). The SVM implemented uses the comprehensive one-vs-one approach in a fast implementation called sequential minimal optimization (SMO). He reports that SVM and NB were the best performers on classification accuracy with SVM performing slightly better with larger training sets. However, his study on training times with respect to training set size showed that Naïve Bayes showed a relatively constant training time whereas SVM initially shows an exponential increase which levels off at a training time which is 100 times that of the Naïve Bayes training time. The classification time (testing time) of SVM also increases greatly with the training set size whereas Naïve Bayes shows small variations. The kNN classifier also has a very high classification time. Thus SVM and kNN do not scale well with increasing training set sizes whereas NB and Rocchio perform well on this metric.

Several researchers have compared various classifiers without specifically targeting the text domain.

Kiang (2003) studied the effects of data characteristics on classifier performances. She performed controlled experiments with synthetic data to observe the effects of various data imperfections. The classifiers used for comparisons were neural networks (multilayer perceptron), C4.5 decision tree, logistic regression, linear discriminant analysis (LDA) with Bayes classification, and the k-nearest neighbor (kNN). The different data characteristics studied were base case, nonnormality, nonlinearity, dynamic scenario, high correlation, multimodal distribution, unequal sample proportion, unequal covariance and sample size. In all cases the neural network method performed the best except for large sample size in which case kNN performed the best. The performance of logistic regression followed that of neural networks with C4.5 and LDA_Bayes being far behind. Sample size had a large influence on kNN and C4.5 which showed significant improvement with increasing sample sizes.

Kotsiantis (2007) presented a review of classification techniques for supervised learning. He compared several techniques on many relevant criteria such as training times, storage requirements, robustness and transparency. According to this paper, kNN requires zero training time while Naive Bayes and decision trees train quickly. The training times of neural networks and SVMs are larger than these by several orders of magnitude. kNN has the maximum storage requirement for both training and testing as it stores all training instances for comparison during testing. For other classifiers memory required during testing is much less than that required during the training phase. Naïve Bayes requires very little memory for both training and testing. Naïve Bayes can handle missing values and along with rule-based and tree-based classifiers is resistant to noise. kNN is most susceptible to noise as noise can easily distort its similarity measures leading to wrong classifications. Transparency is defined as the ease of being understood by human beings. On this index, Naïve Bayes, decision trees, rule-based classifiers are the best followed by kNN which is quite good. SVM and Neural networks are deemed to be the least transparent.

The analysis by Kotsiantis shows that Naïve Bayes has some very desirable qualities i.e. very small training and test timings, low memory storage requirements, transparency, resistance to noise and the capacity to handle missing values. Furthermore, Naïve Bayes can also work as an incremental classifier. Thus we can expect that the application of Naïve Bayes will continue to be explored for text classification despite the earlier negative reports.

Kiang's analysis shows that neural networks are the best classifiers and that increasing the training sample size significantly increases the performance of both kNN and C4.5 decision tree. Since in the text domain, large training sample sizes are easily available, these two classifiers cannot be removed entirely from consideration. One major concern with kNN is the large classification/test time which increases with increasing training data and thus it may not be a suitable classifier especially for web-based applications where speed is an essential characteristic. kNN is also very susceptible to noise. Hence even though kNN has been found to be a very good text classifier in the past, its further application in the web domain may be limited due to the requirement for fast classification and the presence of large training data sets.

Decision trees, on the other hand, require less time for training as well as classification. Therefore we can expect to continue seeing their application in text classification.

2.1.2 Multilevel Data in Today's World

The advent of internet and its increasing popularity has resulted in an overwhelming amount of documents presently available along with a very wide variation in their content. To structure this content for easier accessibility, these documents are often arranged at multiple levels along a concept hierarchy. Hierarchies are not unique to the web. Documents collected for a specific purpose e.g. collection of medical documents (MEDLINE), patent documents (WIPO) and news articles (RCV1) are all structured along a hierarchy. Similarly on the web, Yahoo! and DMOZ are two examples of systems which follow a structured document catalogue. The size and depth of data taxonomies is increasing with the current explosion of data. Taxonomies now consist of thousands of categories. An exhaustive study in 2004 found the Yahoo! directory to contain 292,216 categories in a 16-level hierarchy. Single classifiers discussed in the earlier section do not take advantage of this hierarchical information. The hierarchy has to be flattened to a single level for the application of these classifiers. Flattening results in a huge number of categories which have to be differentiated by a single classifier. Single classifiers are not able to handle such a large number of categories. For example, the time complexity if an SVM is directly proportional to the number of categories. This training time soon reaches unacceptable levels with the amount of categories available in current systems. Furthermore, the information inherent in the hierarchy is lost during flattening and a single classifier is not able to focus on differences between categories at the lower level of a hierarchy. Therefore text classifiers which use this hierarchical information present with the data are needed for further improvement in classification performance.

2.1.3 Hierarchical Text Classification

Koller & Sahami (1997) presented one of the earliest works on hierarchical classification of documents using Naïve Bayes and KDB which is a more complex Bayesian classifier. They used the Reuters 22173 dataset and extracted three hierarchical data sets from it. These datasets consisted of three levels each with different number of features and documents. Text documents were successively applied to classifiers at the first and second levels for a final decision to be made. Errors made at the first level could not be corrected at the second level thus propagating them down the hierarchy. The baselines were the corresponding flat representations of these three sets. They initially applied aggressive feature selection which significantly improved the performance of both the flat as well as the hierarchical classifier. The comparison between flat and hierarchical classifiers was inconclusive with Naïve Bayes. For two hierarchies, the hierarchical Naïve Bayes classifier performed better whereas for the third hierarchy, flat Naïve Bayes performed better. However, hierarchical classifier using the more complex KDB showed considerable improvement over the corresponding flat KDB classifier. With an optimized number of features, the hierarchical KDB showed an 80% reduction in error over the flat KDB for one of the hierarchical datasets. They conclude that the use of a structured topic hierarchy along with the use of aggressive feature selection and more complex classifiers would lead to significantly better classification of text documents.

McCallum et al. (1998) attempted to improve the accuracy of a Naïve Bayes classifier by using class hierarchies of the UseNet and Yahoo datasets along with a set of corporate web pages. They asserted that large hierarchies often have sparse training data per class especially at leaf nodes. They applied a method called “shrinkage” which improves the maximum likelihood (ML) estimate of a leaf node by taking a weighted sum of the ML estimates of all the nodes on the path from the root to that leaf node. They reported a 29% reduction in classification error using this method. They used mutual information to select relevant words for feature selection at each internal node of the tree but their experiments showed that maximum accuracy was achieved with no feature selection. To increase computational efficiency, the tree was pruned dynamically during classification. They showed that pruning improved efficiency with only a slight reduction in classification accuracy. Their experiments also showed that using even a partial hierarchy had a better performance than that of a flat classifier.

Weigend et al. (1999) experimented with a two-level hierarchy on the Reuters-22173 corpus using neural networks. Using cluster analysis, they manually grouped the topics into several meta-topics. They claimed that a large number of wrong assignments in a flat classification model are made on topics which are semantically close to the actual topic and thus fine grained distinctions are necessary to improve classification. In their model, a test document was given to each of the topic classifiers as well as to the meta-topic classifier. The final result was obtained by multiplying the individual topic classifier and meta-classifier outputs. They used two techniques for dimensionality reduction. The first one used Latent Semantic Indexing (LSI). For input to the meta-topic classifier, LSI was done on the entire corpus documents whereas for input to the classifiers on the second level, LSI was done on documents belonging only to a particular meta-topic. The second technique used chi-squared term selection to choose a small subset of important terms for vector dimensions. They reported that the performance of the two-level system was better than that of a flat system especially on lower frequency topics. They experimented with both global and local LSI representations as well as global and local term selection methods and reported that while local LSI outperformed global LSI, global term selection gave the best performance. They achieved a 5% improvement for averaged precision with the maximum improvement on rare topics.

Fukumoto & Suzuki (2002) built a hierarchical classifier using a combination of Naïve Bayes (NB) and SVM on the 1996 Reuters Corpus which has 126 categories in a four-level hierarchy. The one-vs-rest model of SVMs was used for multiway classification. A separate classifier was learned for each internal node of the hierarchy tree. Each classifier thus selected the path to be followed until a leaf node was reached. NB classifiers were trained using 10-fold cross validation. SVMs were trained on the held out test data (or evaluation data) which could not be classified correctly by the NB classifiers. This process was repeated for classifiers at all nodes. During the test phase, first the NB classifier was applied using a threshold determined during the training phase. If the test instance could not be classified by NB, it was given to the corresponding SVM for classification. They compared their NB/SVM combination hierarchy results with those obtained by a hierarchy of only NB and hierarchy of only SVM along with that of flat NB and SVM classifiers. Their results showed that hierarchical NB and hierarchical NB/SVM performed better than the flat classifiers. However, the hierarchical SVM performed worse than the flat SVM with its F1 measure value dropping to half of the flat SVM's F-measure.

Yang et al. (2003) experimented with hierarchical organisations using SVM as well as kNN on the OHSUMED corpus with 14,321 categories on 10 levels. They found that the training time of flat SVM was 102 hours whereas the corresponding time for hierarchical SVM was only 26.3 minutes. In kNN however, the main time complexity is in the classification (testing) phase. This is almost same for each level in the hierarchy and hence this multiplies with the number of levels used. kNN is therefore not suitable for a hierarchical organization. These experiments did not compare the classification performance (accuracy/F-measure) of flat and hierarchical organisations and only show that the time complexity of SVM is improved by using a hierarchy.

In a subsequent paper, Liu et al. (2005) reported an evaluation of a hierarchy of SVMs on the complete Yahoo! taxonomy along with an analysis of the Yahoo!

taxonomy itself. They collected their data in 2004 and at that time the Yahoo! directory had 292,216 categories organized in a 16-level hierarchy. They reported that the category distribution was skewed and that 76% of the categories had less than 5 labeled documents. These categories were called rare categories and their incidence increased at deeper hierarchy levels. The performance of the hierarchical SVM organization was compared with that of a flat SVM on both time complexity and effectiveness (using both micro- and macro-averaged F-measures). Flat SVM was found to be very unsuitable as it took 13 days with a parallel combination of 10 powerful machines for training while the average classification (test) time per document was 0.69 seconds. The best hierarchical combination, on the other hand, took only 2.1 hours for training and an average of 0.0016 seconds for classification. The study of classification performance with respect to the depth of the hierarchy however showed that increasing the number of levels actually decreased the effectiveness. Using categories at the deepest (16th level) for both flat and hierarchical SVMs showed a very low performance though the hierarchical SVM performed better than flat SVM. Even with the best settings, Hierarchical SVM showed a Micro-F1 of 0.24 and a Macro-F1 of 0.12. Liu et al attribute this to data sparseness in the lower level rare categories and suggest that increasing the number of training examples for these categories would improve classification. They compared the performance of the hierarchical SVM with the number of training examples per category and found that increasing the number of training samples per category beyond 100 significantly increased classification performance levels.

Wetzker et al. (2008) studied the effects of taxonomy size on the classification performance of a Naïve Bayes classifier. They used the RCV1 corpus with the *topics* hierarchy which had 104 categories. They applied a greedy algorithm to find an optimal subset of the full category set using four utility measures – utilities by Occurrence, Graph Entropy, SVD and expected F-Measure. They reported that the expected F-measure gave the best performance and that only about 20% of the categories were required to achieve near to optimal classification performance. They concluded that reduction in the number of categories was very important in the face of continuously expanding datasets in current times.

Ghazi et al. (2010) compared a flat SVM with two-level and three-level hierarchy of SVMs for classification of emotions in text. They used two different corpora – blog sentences and children's stories both annotated with emotion labels. They reported the hierarchical approach improved the classification results and was also better at handling data imbalances.

All the above approaches use a divide and conquer strategy with successive refinements and the use of many classifiers at different levels. One major drawback of this method is error-propagation. An error made at any level of a hierarchy cannot be rectified at lower levels. Some researchers have explored alternate methods to bypass this problem.

Cai & Hoffman (2004) presented a method of modifying a multi-class SVM for classification using hierarchy information. A single classifier is learnt instead of multiple classifiers at different levels of the hierarchy. Their experiments were conducted on the WIPO – alpha collection. This is a collection of patent documents structured in a four-level hierarchy and is published by the World Intellectual Property Organization (WIPO). A class attribute representation was used which encoded the relationship between classes in the data hierarchy. A discriminant function was developed which took the contribution of all nodes on the path leading to the leaf node. They reasoned that predicting a class near to the original class was less costly than predicting a class which was far away from it. They defined a loss function based on this idea and the classification process worked on minimizing this loss. They also evaluated their system on the parent accuracy measure and argued that a higher parent accuracy would confine misclassifications into the original category's siblings rather than into far away nodes. This would be useful in automatic systems which are designed to assist human experts. They reported that their hierarchical SVM outperforms the flat SVM on the loss function and in *most cases* performed better on the evaluation measures of accuracy, precision and parent accuracy. The performance gains were higher in cases with fewer training documents.

Qiu et al. (2009) also built only one SVM classifier for the entire hierarchy. They used the global margin maximization method which attempted to separate all nodes in a hierarchy from their sibling nodes. They tested their model on the WIPO-alpha collection which consists of 1372 training and 358 test instances with 188 categories divided into 3 levels. Again, the classifier used was a multi-class SVM. They compared their implementation with two other hierarchical models based on a single SVM and showed that in their system the hierarchical losses were reduced.

Gao et al. (2009) proposed a classifier independent framework to deal with the problems of data skew and error propagation in earlier hierarchical classification methods. This framework consisted of two stages – one to limit errors at the current level and the other to correct errors made by a previous level. The dataset used was the top 10 levels of the ODP web pages dataset. They developed a path semantic vector which incorporated the semantic information from a category hierarchy. They introduced category probability and subtree probability as means to reduce classification errors at higher levels. Correction of classification errors was done using co-occurrence probability which was the probability of any two given categories being assigned to a document. The prior information was generated offline. They tested their architecture using the SVM and Bayesian classifiers and showed performance improvement over the corresponding classifiers. They compared the classification performance across different levels of the hierarchy and show that more improvement occurred at lower levels. The evaluation metric used was the micro F-measure.

Chuang et al. (2000) focused on the speed of hierarchical classification and conducted their experiments on a collection of web pages consisting of 200 news items on professional baseball and basketball. They used a concept hierarchy where each node was represented by a TFIDF feature vector. These TFIDF feature vectors were derived from the collection of all documents belonging to that node. The documents set of a parent node was taken as the union of the document sets of its child nodes. During the training period, a threshold distance was calculated for each node within which a document was considered to belong to the node’s category. During the testing phase, a node’s TFIDF vector and threshold together acted as a classifier. A new test document started its comparison with the root node using the cosine measure. If the cosine distance was less than the threshold, the new document was considered to belong to that node’s category and was filtered down to the child nodes for further comparison. A document was assigned all categories whose node TFIDFs fell within the corresponding threshold values. The number of filtered documents reduced with the depth of the hierarchy. Accuracy was calculated as the fraction of the correctly classified documents out of the total number of documents filtered to that node. The results showed an accuracy around 90% at the top two levels and a progressive drop in accuracy from level 3 (75%) to level 5 (45%). They further experimented with reducing words in the feature vector by retaining the top TFIDF feature values and reported that a 40-50% reduction in features hardly affected classification accuracy. They also experimented with a method of introducing background knowledge and showed that it improved classification accuracy. They further applied this method to TV closed caption data along with 15% web pages and showed promising results. TV captions can thus be used in video and multimedia classification. Their timing analysis showed a low training time of O(n log n) where n was the total number of documents.

2.1.4 Conclusion

A study of traditional text classifiers shows that kNN is a very good classifier and that SVM is the best text classifier in a binary setting. However document classification often involves many classes or categories. This led to the development of the multi-class SVM whose time complexity was proportional to the number of classes present. This caused the training time of SVMs to rise rapidly. kNN is a lazy classifier which stores all training instances for comparison during the testing phase. This causes a high test time complexity which soon becomes impractical with large training sets. The Naïve Bayes classifier which was earlier reported not to be suitable for text classification is seeing a renewed interest due to its very low training/test timings and other beneficial characteristics. Decision trees also have low training and testing times and continue to be used in classification systems. Neural networks are scalable and have low classification (test) times. In the present day, classification speed is an important aspect of document classification along with classification accuracy. Thus, there is no clear winner on both counts in a multi-class classification setup.

In the present day, the web has resulted in a huge amount of data along with taxonomies consisting of thousands of categories. Traditional single level classifiers are now unable to handle the extent and complexity of this data. Classifiers are which utilize this hierarchical information are thus required to further improve classification performances. Literature shows that the use of existing topic hierarchies can drastically improve the timing efficiency of a classifier and also improve classification accuracies especially of rare classes. In a hierarchical classifier, individual classifiers deal with smaller datasets, less categories and a lower number of features. This improves the efficiency of the concerned classifier and also improves fine grained distinction between closely related topics. However using many levels of hierarchy actually degrades performance. The main reason for this is error propagation where errors made at higher levels of the hierarchy cause further errors at lower levels. Another reason is the sparse number of training documents at lower levels. Hierarchical representations with the SVM classifier have been studied extensively but have not shown any major advantages over hierarchical representations with other classifiers such as Naïve Bayes and Neural Networks. On the other hand, the kNN classifier has been shown to be totally unsuitable for a hierarchical setting.

A few researchers have proposed implementations of a single SVM multiclass classifier to deal with hierarchical information. However these studies were conducted on a very small dataset with few levels of hierarchy. This method seems inappropriate for scaling to very large datasets with large number of categories and many hierarchy levels. A method for error reduction and correction has also been proposed for a hierarchical arrangement of classifiers. This method involves a lot of prior information and computations at each node which would affect training times adversely.

This suggests that the popular divide and conquer strategy of text classification with the use of successive classifiers at different levels along with feature reduction is best suited for scaling to a large number of documents as well as to a large number of categories. Training time is also significantly reduced in this method. To reduce the effect of error propagation, a small number of levels (two or three levels) should be used. These levels need not be the top two/three levels. The given hierarchy structure can be optimized by removing some intermediate levels between the root and the leaf nodes to create a two/three level category hierarchy. Unlike the case of SVM for binary text classification, the current research on hierarchical text classification does not throw up any specific classifier as a clear winner. As such, an extensive experimental study involving classifiers of different kinds is needed to assess whether any particular type of classifier is more appropriate for exploiting the concept hierarchies inherent in today's datasets and which classifiers, if any, benefit more from a hierarchical organisation.

2.2 Subspace Learning in Multi-Level Text Classification

As the volume and diversity of data increases, the number of dimensions required to represent the data also increases drastically. Such high dimensions adversely affect classifier performances. Subspace Learning is a technique used in many fields to bring down the number of dimensions. Application areas of subspace learning include image processing, pattern recognition, computer vision, robotic vision, human gait analysis, object classification, document classification and multimedia classification to name a few. Research in subspace learning is broadly divided into two main areas – *Feature subspace learning* which focuses on finding a reduced set of dimensions to represent the entire dataset and *Data subspace learning* which tries to find an optimal data subspace along with features corresponding to that subspace to improve overall classification performance.

2.2.1 Feature Subspace Learning

Linear Discriminant Analysis (LDA) (Fukunaga, 1990) and Principal Component Analysis (PCA) (Joliffe, 1986) are two traditional methods of feature reduction. LDA is a statistical method of transforming a high dimensional space to lower dimensions. It uses the class information present in the data and is thus a supervised method. PCA, on the other hand, is an unsupervised method which is very popular in the field of pattern recognition and computer vision.

Szepannek and Luebke (2004) introduced the concept of *characteristic regions* and presented their Different Subspace Classification (DiSCo) method to simultaneously visualize as well as classify multiple categories in presence of high dimensions. They used the IRIS dataset and compared their method with CART decision trees and LDA showing it to outperform CART and approaching the performance of LDA.

Li (2004) proposed an incremental version of PCA to deal with large datasets. He claimed that it was better suited to real-time applications than the computationally expensive standard PCA. He applied it successfully to dynamic background modeling and multi-view face modeling showing its superiority to standard PCA.

Cao et al. (2007) presented a method using a subset of the kernel space to extract the most informative features for classification. They used the IDA Benchmark repository (medical diseases data) with the SVM and kNN classifiers. They showed that their method had lower computational complexity compared to the baselines (Generalized Discriminant Analysis(GDA) and Kernel Fisher Discriminant Analysis(KFD)) and that the classifier performance of SVM was better than that of kNN.

Chen et al. (2008) constructed an optimal subspace kernel with an eigenvalue solution. They proposed a method to simultaneously learn the kernel subspace as well as the kernel classifier and showed that it was effective. The SVM classifier was used with the seven datasets. Five datasets were from the UCI Machine Learning Repository (satimage, waveform, segment, wine, and USPS) while two were gene expression data sets.

Cohen and Paliwal (2008) constructed a subspace for each class using class dependent PCA. They implemented the nearest subspace classifier and compared it with the nearest neighbor and the nearest centroid algorithms on microarray cancer data showing that it performed better than the baselines.

He and Cai (2009) proposed an active subspace learning algorithm for relevance feedback driven image retrieval. Each iteration of the algorithm presented the user with five images for labeling. The user provided labels were then used for tuning the system. Four iterations were used for each query. They used 7900 images from the COREL dataset and compared their algorithm with two other active learning algorithms, a semi-supervised algorithm and the SVM which is a passive learning algorithm and showed their algorithm performed the best.

Yang et al. (2009) proposed a ubiquitously supervised subspace learning prototype to deal with image misalignments in computer vision. They claimed that most of the existing supervised as well as unsupervised subspace learning algorithms could be considered as special cases of their prototype. This prototype was then used to generate misalignment-robust versions of PCA and LDA along with two other feature reduction techniques (MFA and NPET). They used the CMU PIE (Pose, Illumination and Expression) database and the FRGC image database with the nearest neighbor classifier for their experiments. Their results showed that all the misalignment-robust versions perform better than their original counterparts for face recognition.

Kwak and Lee (2010) proposed their own versions of PCA and LDA called WPCA and LDAr to extract features for regression problems. They compared the performance of these with some other feature extraction methods on the Housing dataset from the UCI machine learning repository and the Orange juice dataset from the UCL machine learning database using the weighted kNN regressor with k=5. They reported that while WPCA performed slightly better than PCA, LDAr outperformed all other methods.

Yaslan and Cataltepe (2010) used the mutual information between class labels and features to produce relevant random subspaces for semi-supervised ensemble learning. They compared their method with random subspaces on five datasets (three datasets from the UCI machine learning repository along with a text dataset and an audio genre dataset). The classifiers used were kNN, Linear Bayes and J48 (C4.5). Their experiments showed that their method performed significantly better in the presence of many irrelevant features, a smaller ensemble and a smaller number of features.

Hu et al. (2010) suggested a spatiotemporal subspace learning algorithm for gait recognition. Unlike the two-dimensional image data, gait data has third time dimension. They proposed a gait feature vector (GFV) using PCA and Discriminative Locality Alignment (DLA) and used the USF HumanID gait database for their experiments. They reported that their vector showed good discriminative power for recognition of individuals.

Chatpatanasiri and Kijsirikul (2010) presented a general feature reduction framework for semi-supervised learning using the nearest neighbor classifier and apply it to three subspace learning algorithms. Their experiments were conducted on the Ionosphere (radar pulses), Balance (psychological), BCI (EEG graphs), USPS (handwritten digits) and M-Eyale (face recognition) datasets. They asserted that their algorithms achieve very good performances for the semi-supervised setting.

Calabuig et al. (2010) proposed a Fast Hopfield Neural Network (F-HNN) using subspace projections. It confined the direction of movement of the neural network to a subspace of constraints. They compared their F-HNN with two other HNNs on the N-queens problem with N=16 and showed that their F-HNN was 20 times faster than the two baselines.

Xu et al. (2011) proposed a fast kernel subspace learning method (TAKES) and conducted their experiments on medical data using the kNN classifier. They showed that their method was faster than the other kernel methods.

Other application areas of feature subspace learning include Multispectral Remote Sensing Image Classification (Bagan & Yamagata, 2010), Robot Vision (Nayar, Nene, & Murase, 1996), Tensor Data Learning (Lu, Plataniotis, & Venetsanopoulos, 2011), Face Recognition (Liu, Chen, Zhou, & Tan, 2007), Image Segmentation (Law, Lee, & Yip, 2010), Multiple Feature Fusion (Fu, Cao, Guo, & Huang, 2008) and Music Genre Classification (Panagakis, Benetos, & Kotropoulos, 2008), (Chen, Gao, Zhu, & Sun, 2006)

2.2.2 Data Subspace Learning

A lot of research in this area concentrates on subspace clustering (also called projected clustering). Subspace clustering tries to find clusters present in different subspaces of a dataset. It is therefore a combination of a *search method* (to find the subspace) and a *learning method* (to find the clusters within the located subspace).

Parsons et al. (2004) categorised subspace methods into two groups – *top down* and *bottom up* based on search methods. The *top down* approach starts with the complete feature space where all dimensions have equal weights. An approximation of the clusters is made at this stage and each dimension is weighted for different clusters. At each subsequent iteration, clusters are regenerated using the updated weights. Top down approaches generate disjoint partitions of similar sizes. Bottom up algorithms, on the other hand, start by creating histograms for each dimension and then selecting only those dimensions with densities above a threshold. Clusters are then formed by combining dense units. Bottom up approaches can generate overlapping clusters. They compared the performance of a top-down algorithm (FINDIT) with a bottom-up algorithm (MAFIA) on synthetic datasets. They show that while MAFIA outperformed FINDIT on smaller datasets, the top-down FINDIT eventually outperformed the bottom-up MAFIA on huge datasets.

Wang et al. (2004) presented a grid density based subspace clustering algorithm. They used Region quadtrees which are spatial data structures that use binary subspace division. Two dimensional clustering was implemented using quadtrees. They asserted that high density units represent cluster centers. Their algorithm repeatedly divided the non-empty data spaces. Empty data spaces were not processed resulting in higher speeds. They compared their algorithm with the K-Means algorithm on a synthetic dataset as well as the KDD CUP’99 dataset and showed it to give better results.

Yiu and Mamaulis (2005) proposed the branch and bound method to find projected clusters by mining frequent itemsets. They used synthetic as well as real world data (Image Segmentation Data from the UCI Machine Learning Repository and the BioID Face Database). They compared their methods with some existing projected clustering methods and show that their method was faster and produced clusters of high quality.

Moise et al. (2006) proposed an algorithm called P3C – Projected Clustering via Cluster Cores. They started by defining cluster cores which were areas containing a very high number of data points. These cores were then repeatedly refined to produce the final subspace clusters. They used the EM algorithm to calculate membership for each data point. Their experiments were run on synthetic data as well as real world cancer data and housing data sets. They compared their results with those some other subspace clustering algorithms and showed them to be better. A further comparison with the full space clustering produced by the k-means algorithm showed that full space clustering did not produce the same clusters.

Zaki et al. (2007) presented an algorithm called CLICKS which produced subspace clusters for discrete valued data. They claimed that most of the earlier subspace clustering techniques worked only with numeric data. They represented the discrete valued dataset as a k-partite graph and used the strongly connected property of graphs to find the set of all k-partite cliques. Subspace clusters were taken as all cliques where $k < n$ (total number of data dimensions). They used synthetic datasets to compare their algorithm with a few other subspace clustering methods which work on discrete-valued data and showed their algorithm to be superior. They then applied their algorithm on two real world datasets – the Mushroom dataset and the Congressional votes dataset from the UCI machine learning repository and showed that their subspace clustering results were better than full dimensional clustering results

Zhou et al. (2008) explored the use of a Bayesian Network for projected clustering. They proposed that adjacent cells identified by a Bayesian network could be merged together to form a projected cluster. They performed their experiments on the Chest-Clinic dataset of 1000 records and showed that their method was feasible because it gave the same number of subspaces as detected by a traditional density-based clustering algorithm.

Boutemedjet et al. (2010) presented a subspace clustering method for non-Gaussian (non-normally distributed) data. They applied minimum message length, which is a data compression technique, to model selection. They defined a message length objective function to select both the subspace as well as the subspace features. They evaluated their method on a dataset containing images from eight categories. They compared their method with a clustering method on non-Gaussian data without subspace clusters and showed it to be superior.

Hotho et al. (2001) claimed that clusters which are deemed to be of good quality based on statistical measures are often not suitable for real world applications such as document search in a large dataset. They further claimed that the usefulness of document clustering depends on the user’s view i.e. different documents may fall in different categories depending on the task for which they are required. They proposed the incorporation of background knowledge or ontology in the preprocessing step to produce useful clusters and showed that preprocessing based on different ontologies produced different clustering outputs. Their method of preprocessing was based on concept selection and aggregation (COSA). They claimed that k-means clustering based on COSA produced better clusters than a baseline k-means clustering which used tf-idf with the top ***d*** terms.

The incorporation of domain knowledge in subspace clustering has also been explored. Liu et al. (2004) developed a framework to generate clusters involving ontology information. They proposed an ontology relevant cluster tree using ontology based pruning. They used the Gene Ontology (GO) to cluster biologically related genes. They claimed that their ontology based clustering outperformed normal clustering and ontology based pruning reduced the search space of the clustering algorithm.

2.2.3 Subspace Learning in the Text Domain

An early paper by Schutze and Silverstein (1997) stated that standard clustering algorithms were too slow for real-time applications. They proposed projection via truncation of the feature vector as a method of speeding up the clustering process. Using a subset of the TREC-4 text collection, they compared the performance of a centroid based clustering algorithm with and without projection using both LSI and Term Frequency and reported that truncating upto 50 terms had no effect on cluster quality. They also reported that clustering after projection was much faster than full data clustering.

Torkkola (2001) reported the first application of Linear Discriminant Analysis (LDA) to document classification using the Reuters-21578 dataset and the SVM classifier. He argued that LDA was better suited to document classification than PCA, another dimensionality reduction method which ignores class labels. His results showed that reducing the number of feature dimensions from 5718 (complete set) to just 12 actually reduced the error rate from 11.2% to 8.9% while also reducing the computation time to one-fifth of the full feature computation time.

Yan et al. (2005) presented an incremental supervised subspace learning algorithm called IIS to optimize the inter-class scatter. They used the SVM classifier to show that their algorithm performed better than the baselines (Incremental PCA and Information Gain). The experimental dataset was the Reuters RCV1 Corpus using the four highest topic codes.

Jing et al. (2005) presented their own feature weighting K-Means algorithm for subspace clustering. They assigned large weights to features that formed the subspaces and used the 20-Newsgroups text data to evaluate their algorithm. They compared their algorithm with standard K-Means and bisection K-Means and showed it to be superior.

Yan et al. (2006) proposed a scalable algorithm called Supervised Kampong Measure (SKM) which assigned each data point close to its class mean while maintaining the maximum distance from other class means. Their experiments were performed on the complete Reuters RCV1 dataset using the four highest topic codes with the SVM classifier. They evaluated performance using the F1 measure and showed that their SKM performed much better than the incremental PCA and Information Gain (IG) methods. They asserted that it is very hard to compute LDA on the complete Reuters dataset and thus it was not used in the comparison. For comparison of SKM with LDA and PCA, they used six subsets of the UCI machine learning repository using the kNN classifier and showed that SKM was better than both LDA and PCA.

Salakhutdinov and Hinton (2009) presented Semantic Hashing as a way to speed up document retrieval using the 20-Newsgroups and the Reuters RCV II datasets. They used 128-bit codes to map similar documents to nearby memory addresses. Semantic Hashing mapped documents directly to the hardware, making retrieval very fast and independent of the size of the document set. They also used semantic hashing to extract a relevant subset of documents for TF-IDF calculation. They reported that applying TF-IDF to a subset of documents gave higher accuracy than applying TF-IDF to the full dataset. They used the Restricted Boltzmann Machine (RBM) for their experiments.

Gangeh et al. (2010) applied the Random Subspace Method (RSM) to text classification using a subset of the Brown Corpus. The feature space was very large (22244 terms) compared to the number of documents (495). They performed feature selection in two steps – first by document frequency (DF) and then by Information Gain (IG). They showed that feature reduction upto 25% had no effect on classification performance. The Random Subspace Method works by randomly dividing the feature space into subspaces and applying the base classifier to each subspace. The final classification result is obtained by combining the outputs of all the base classifiers usually by a majority vote. The performance of their RSM method using SVM as the base classifier was compared to single SVM and kNN classifiers on both full and reduced feature spaces. The RSM method using SVM performed the best followed by SVM with two-step feature reduction. kNN was the worst performing classifier here.

2.2.4 Conclusion

A majority of the subspace learning work in Pattern Recognition, Image Processing, Computer Vision, etc deals with the creation of feature subspaces with transformed features which can be used to distinguish between objects in the complete data space. While these methods might be appropriate for the pattern/image domain with high dimensions but few categories, they would not be suitable for the currently emerging web text data with a very large number of categories. In the text domain, we now need to differentiate between similar subcategories within a larger category. As such, we need to focus more on smaller differences which would not be possible with a reduced feature set on the complete data. Subspace Clustering, which works by extracting a data subspace along with features relevant to that data subspace, intuitively seems to be more appropriate for our problem domain of text with a large number of categories. A number of researchers have applied subspace clustering to uncategorised documents. However there are large datasets available with associated category information. This information should therefore be used for text classification. The current state of subspace research indicates that classification is more popular with feature subspace learning while clustering is more popular with data subspace learning. The analysis of our problem domain, however, suggests classification to be more appropriate due the presence of class labels. There is therefore a need to develop classification methods which work on data subspaces. While two-level hierarchical classification may seem analogous to this situation, a major difference is the use of a classifier at the top level to detect the first level of categories in hierarchical classification. Data subspace methods, on the other hand, use search techniques to detect subspaces. This is very useful for the speed of classification/retrieval. Hence the need of the hour is to have a search based method to detect the subspace (first level category) followed by *classification* within the subspace with reduced dimensions to detect the second level of categories. The study of hierarchical classification earlier has shown that error propagates down the hierarchy and that methods of error correction are computationally very expensive. In order to keep retrieval speeds high, error correction has to be avoided. The number of levels has to be kept small (2-3 levels) to minimize error propagation. Thus a two-level classification system with a search based method to detect categories at the first level seems to be most appropriate for classification of text documents with large taxonomies. Literature has also shown SVM and kNN to be the most widely used classifiers in subspace learning. Our earlier analysis on text classification has however shown that while SVM is the best binary classifier, the same does not hold true for multi-class problems and that kNN is not suitable at all for large training sets as it postpones all computation to the run-time classification phase. Therefore, a comparative analysis of many types of classifiers is required to determine which classifiers are best suited to subspace classification. Furthermore, as preprocessing based on ontologies has been shown to produce better clusters, the use of background knowledge in the search method for the detection of appropriate subspaces (first level categories) should also be explored.

2.3 Classifier Combinations in Multi-Level Text Classification

The combination of two or more classifiers has been used in many fields to improve classification results. Classifiers combinations can be of two types – classifier ensembles and parallel classifiers. In classifier ensembles, a number of classifiers of different types are applied on the complete dataset using full feature space and the final result is obtained by combining the results of the individual classifiers. Parallel classifiers, on the other hand, are combinations of classifiers which work in parallel on different portions of the feature/data space. Parallel classifiers can be either of same or different types.

2.3.1 Classifier Ensembles

Classifier ensembles are based on the reasoning that strengths and weaknesses of various classifiers can compensate each other. An instance which is misclassified by one classifier may be classified correctly by another classifier thus pushing up the combined classification performance. Combining diverse classifiers is an essential characteristic for a successful combination. Kittler et al. (1998) state that different classifiers within a combination should never agree on a misclassification i.e. the same incorrect class should not be assigned to a test instance by two or more constituent classifiers. Different classifiers can be trained either by using different input representations for the data, different parameters for the same type of classifier (e.g. different k values for the kNN classifier; different weights for an MLP classifier) or different classifiers altogether (e.g. Naïve Bayes and Decision Trees). There are also a number of rules for combining the outputs of various classifiers within a combination. The most popular one is the majority vote rule where the category receiving the most votes is assigned as the category for the test instance. Other rules are based on mathematical functions such as product, sum, min, max and median.

Xu et al. (1992) applied classifier combinations for the recognition of handwritten numerals. They defined three levels of output information produced by single classifiers – abstract level (a unique label), rank level (a ranked list of all labels) and measurement level (a numeric value for each label showing the test instance's degree of association with that label). They asserted that the highest information was contained in the measurement level. Depending on these levels, they defined three types of classifier combinations: Type 1, Type 2 and Type 3 which combined the output information in the abstract, rank and measurement levels respectively. They stated that the measurement level could be used to combine classifiers such as Bayesian, k-nearest neighbor and other distance-based classifiers. They suggested two versions of voting for Type 1 combinations. They used 6000 handwritten samples from the U.S. Zipcode database and four classifiers using different types of features extracted from the handwritten numerals. Their results showed that the combination of several classifiers had a much better recognition performance than the corresponding single classifiers. They concluded that the focus of future research should change from building a single good classifier with input feature reduction to building a number of classifiers using different and complementary vectors of low dimensions.

Kittler et al. (1998) also tackled the problem of handwritten character recognition using classifier combinations. Scanned images of single handwritten numeric digits served as the data. The four classifiers used were Hidden Markov Model (HMM), Neural Network, Gaussian and Structural classifiers. Four different representations of the characters were used as input for the four classifiers. The outputs of these classifiers were then combined with different combining rules. Their experiments showed the *sum* rule and the *median* rule to be the best performers which were closely followed by the *majority vote* rule.

These rules performed better than the best individual classifier (HMM). The *max* rule performed better than all the other classifiers (except HMM). They also performed an error sensitivity analysis and showed the *sum* rule to be most resilient to errors.

Duin and Tax (2000) conducted a large set of experiments with various types of classifier combinations using different classifiers as well as different feature sets along with many types of combining rules. Their data consisted of six different feature sets for 2000 handwritten numerals. Their classifiers included two Bayesian classifiers, three nearest neighbor classifiers, a decision tree, two neural networks and two versions of support vector machines. Their results showed that combination of different feature sets on one classifier was much better than the combination of different classifiers on one feature set and that the best performance was observed by the combination of both different feature sets *and* different classifiers.

Al-Ani and Deriche (2002) experimented with classifier combinations for the classification of texture images and speech segments and also for speaker identification. They used N different feature sets to train a single type of classifier – thus generating N different classifiers. The Neural Network classifiers were used in these experiments. Each classifier produced an output vector with K components showing the degree to which the input vector matched the K label categories. They used the Dempster-Shafer (D-S) theory of evidence which can represent uncertainties to produce a numeric value representing belief in a class label *k* produced by a classifier c_n. The combination method then combined these belief values. Their results showed that combined classifiers worked better than single classifiers and that their combination technique was superior to the other standard techniques. However, their technique was also very computationally expensive.

Jordan and Jacobs (1993) introduced a tree structured architecture called the ***Hierarchical Mixture of Experts*** for combining results of several classifiers. The tree contained gating networks at non-terminal nodes and the classifiers or *experts* at the leaf nodes. The input test vector was applied to all classifiers which produce corresponding output vectors. The output vectors were blended with the gating network outputs to produce weighted outputs of experts which then proceeded upwards in the tree. The root level gave the final output of the classification system.

2.3.2 Parallel Classifiers

Parallel classifiers are based on the premise that reducing the number of features and the data variation that a single classifier has to handle can greatly improve its performance and that this would lead to enhanced overall performance. Duin (2002) presented a theoretical discussion on classifier combinations and suggested that the use of a combining classifier would be better than the use of fixed combination rules such as sum, product, etc. He suggested that the training set could be used to partition the feature space into different regions and decide the base classifier for each region. For each test instance, the combining classifier would have to find its relevant region after which actual classification would be done by the base classifier chosen for that region. He suggested that the training set should be divided into two parts – one for training the base classifier and one for training the combining classifier.

Tulyakov et al. (2008) presented a review of classifier combination methods from the perspective of pattern classification. They categorised combination methods according to complexity, output, etc. They asserted that while there were many methods of generating different classifiers such as the use of different training sets, different input feature vectors, random feature subsets and different initialisations, *the ideal method would be to partition the feature space into regions related to different categories*. They concluded that most classifier combination research was limited to using low complexity combinations such as the sum rule, majority voting, etc. They suggested the use of more complex combination methods and the *use of locality property of classifiers* for future research.

Qi et al. (2011) applied the locality property to develop a locally sensitive SVM (LSSVM) for image retrieval. They applied locally sensitive hashing (LSH) to divide the whole feature space into a number of regions and constructed a local SVM on each of the regions. They asserted that local regions had smaller within-class variance. This corresponded to higher between-class variance leading to easier separability of classes. Thus, in local regions, simple classifiers could achieve a performance comparable to the more sophisticated classifiers but with faster speeds. Their experiments showed that their locally sensitive SVM outperformed the simple full space SVM. Their algorithms were compared on a real-world image dataset collected from Flickr.com

A number of researchers have used a random selection of features to create feature subsets. Ho (1998) used a pseudorandom method to select of subset of feature dimensions. A decision tree was then constructed using the complete training set with only the selected dimensions. Multiple trees were thus constructed using different random feature subsets. A test instance was given to all the trees for classification and the individual classification decisions were then combined to generate the final result. They claimed that their method could take advantage of high dimensions. The experiments were conducted on 14 datasets from the UCI machine learning repository. Their results were better than the results obtained by using a single decision tree of the same type with complete data and full feature set.

Kotsiantis (2009) presented a Local Random Subspace Method to generate localized decision stumps. The experiments were carried out on 27 datasets from the UCI machine learning repository. Decision stumps are one level decision trees that classify instances based on a single feature value. Pseudorandom subsets of the original feature vector were used. Local learning was done by only taking the training points which were close to the test point. These were identified by using the euclidean distance metric. Fifty neighbouring points were used in this case. A random feature vector was then generated and a decision stump trained using these points. The final prediction for a test instance was based on the averaged result of all decision stumps. They showed that their technique worked better than normal K-nearest neighbor method with k=3 as well as k=50 in addition to being better than other methods using decision stumps such as Bagging, Boosting and Multiboost.

2.3.3 Classifier Combinations in the Text Domain

Larkey and Croft (1996) experimented with different combinations using three classifiers: K-nearest neighbor, relevance feedback and the Bayesian Independence Classifier. The single classifiers were used as baselines. The text corpus was a collection of patient discharge summaries with 15 ICD9 codes. Their results on various two-classifier combinations as well as the three-classifier combination showed that all the combinations performed better than the individual classifiers and that the three-classifier combination performed the best. On the measure used for combination, normalized scores performed better than label ranks.

Ruiz and Srinivasan (1999) used the hierarchical mixture of experts to implement a text classification system. They used backpropagation neural networks to implement experts as well as gates and evaluated their system on the UMLS metathesaurus and the OHSUMED test set. The document vector was applied to each of the experts as well as each of the gates. In their system, the gates represented high level concepts and gate outputs were set to 1 or 0 depending on whether the document had the corresponding gate's concept or not. They compared their system with flat neural networks as well as the

Rocchio classifier. They showed that the hierarchical neural structure performed better than flat neural network which in turn performed better than the Rocchio classifier.

Estabrooks and Japkowicz (2001) also presented a text classification scheme based on hierarchical mixture of experts. They designed it to deal with class imbalances and tested it on the Reuters-21578 corpus. They proposed a three level tree with individual classifiers at the leaf nodes, combining experts at the internal nodes and output level at the root node. The two combining experts were the oversampling expert and the undersampling expert which combined the results of 10 oversampling and 10 undersampling classifiers each. They compared their results with those of C5.0 with Adaboost and showed that their scheme was more effective on both precision as well as recall.

Al-Kofahi et al. (2001) applied classifier combinations to the problem of text classification using the American Law Reports (ALR) consisting of 13,779 articles. They presented a Case Routing Program (CARP), a multi-classifier system consisting of two distance based classifiers and two probabilistic classifiers. They used article text and metadata to generated different representations of an article. They also experimented with the use of words, bigrams, nouns and noun-word pairs at the feature level and showed the last two (nouns and noun-word pairs) to be the most effective. They asserted that their system was deployed in January 2001 and that its performance was comparing favorably with the previous manual system.

Florian et al. (2003) presented a classifier combination using four diverse classifiers for named entity recognition in both English and German language data. The classifiers used were Robust Risk Minimization (RRM), Maximum Entropy (ME), Transformation-Based Learning (TBL) and Hidden Markov Model (HMM). The outputs of these classifiers were combined using equal voting as well as three variations of weighted voting. The use of RRM as the combining classifier was also studied. In all types of combination methods, the combined classifier performed better than the best individual classifier on the F-measure metric. The RRM classifier also showed a good performance as a combining classifier. Their results were much better on the English data than on the German data.

Fradkin and Kantor (2005) used three classifiers (Bayesian Logistic Regression, kNN and Rocchio) with normalized scores and different combining rules on the Reuters RCV1 corpus. They used the T11SU (TREC-11) performance measure for comparison. On this measure, their experiments showed Bayesian-kNN and Bayesian-Rocchio to perform better than the 3-classifier Bayesian-kNN-Rocchio combination and the other 2-classifier combinations including kNN-Rocchio. They assert that as kNN and Rocchio are similar classifiers, their results confirm the fact that combining diverse classifiers results in better classification. Their experiments with a variety of combination methods were inconclusive and showed variation with different performance measures.

2.3.4 Conclusion

The origins of classifier combination research seem to be rooted in pattern recognition where the problem domain consists of high dimensions but small number of categories. As such most of the methods discussed are ensemble methods where each classifier is applied on the full data space. As discussed earlier, this is not suitable for the current text domain with a huge number of categories at single and multiple levels. Parallel classifiers, on the other hand, suggest a separation of feature/data space along with the use of a separate classifier for each subspace. The locality property of classifiers enables them to distinguish between small variations among topics within a subspace. As such parallel classifiers are more suited to the task of multilevel text classification than classifier ensembles.

The use of feature subsets has been proposed in literature to reduce classifier complexity and training time. However there are no clear pointers as to the way the original feature space can be partitioned. Random projections of the feature space have been used but there is no guarantee that these projections contain necessary distinguishing characteristics. A partitioning of the feature space which corresponds to a partitioning of the underlying data is logically required. Tulyakov et al. (2008) have suggested that the ideal method would be to partition the feature space into regions related to different categories. This suggests that category information should be incorporated into feature vectors. Thus we have to look beyond the standard tf-idf vectors and even beyond simple semantic enhancements such as clubbing together of similar words based on some dictionary/thesaurus. A category based vector system would be further useful to accommodate the inherent category structure of the data and thus add useful semantic content to the vector representation. Positioning similar categories close together in the feature space can lead to a spatial representation of the category hierarchy within the feature vector. This would enable different types of partitioning to access different levels of information. Thus a parallel classifier system along with the use of a category based vector representation is needed to tackle today's problem of multi-level data classification.

Literature has not identified any specific classifier or set of classifiers which perform well in classifier combinations. The only fact that has come up is with reference to classifier ensembles. This states that the classifiers should be *diverse*. However, there has been no study across a wide variety of classifiers to determine which types of classifiers, if any, benefit more from being in a combination. Most of the work with parallel classifiers has been done using a single type of base classifier. Nowadays, there is a huge amount of data which can be divided into very diverse subspaces. These subspaces have widely differing characteristics. It is quite possible that best classifiers for different subspaces might belong to different *types.* Therefore there is a pressing need to conduct a wide ranging evaluation of hybrid classifier combinations to determine which types of basic classifiers are best suited for parallel combinations in the problem domain of multi-level text classification.

As the web data is continuously expanding, text classifiers need to be retrained regularly to keep up with the increasing variation. As such, classifier training times will play a prominent role in their usefulness. The speed of search/classification is also a very important requirement today. Memory requirements during the classification phase also need to be kept in mind for real-time applications. A tradeoff between classification effectiveness and timing/memory efficiency has to be considered to determine the best solution for our problem domain.

2.4 Chapter Summary

In this chapter we looked at various methods of single level text classification and methods of extending them to deal with multi-level text classification. A survey of basic classifiers showed that the Support Vector Machine (SVM) and the k-nearest neighbours (kNN) were designated as the best text classifiers. The SVM was undoubtedly the best binary classifier but it could not be directly applied to a multi-class setting which is the normal case in text classification. The k-nearest neighbours (kNN) method postponed all computation to classification run time and as such was not suitable with large training sets. Hierarchical classification schemes showed that using the complete hierarchy was not suitable for classification and that the use of 2 – 3 levels was sufficient to improve classification and minimize the error-propagation effect. Subspace Clustering with its use of a data subspace and reduced data dimensions seems quite relevant to multi-level text classification. However, unsupervised clustering can be replaced by supervised classification here to take advantage of existing category information. Hence subspace classification with a search method at level 1 to detect the subspace seems quite relevant to our problem. The use of classifier combinations which are quite popular in other domains was also explored. Parallel classifiers, where a number of classifiers work on different data/feature subspaces, can be applied to multi-level text classification. Feature splits which mirror data splits in the category hierarchy are indicated as an area of future research. Thus a document vector scheme which incorporates the category hierarchy within it needs to be explored. Overall the best methodology seems to be a two-level scheme with a search based method operating on a single document vector to detect the first level category or the subspace. Hybrid combinations using separate classifiers for separate subspaces should be explored. A comprehensive study comparing various types of classifiers for their suitability to multi-level text classification should also be carried out.

Chapter 3

Architecture & Methodology

In this chapter, we introduce the novel techniques of Maximum Significance Value, Conditional Significance Vectors and Hybrid Parallel Classifiers along with the experimental methodology. The explanation is divided into two distinct phases which reflect the sequence of development of the novel techniques and their experimental confirmation. Phase I of the explanation discusses the concept of semantic subspace learning. It presents Maximum Significance Value as a technique for fast detection of the semantic subspace along with a novel vector representation, the Conditional Significance Vector, which enhances the distinction between subtopics within a subspace. It also introduces an initial architecture based on these concepts and explains the experimental methodology used for testing this initial architecture. Once these concepts have been understood, we proceed to Phase II which presents our final Hybrid Parallel Classifier Architecture based on the concepts developed in Phase I. In Phase II, we also present the experimental methodology for testing this final architecture. Towards the end of this chapter, we discuss the performance metrics and the statistical significance tests used to evaluate our final architecture.

3.1 Phase I: Semantic Subspace Learning

In this section we look at the presence of semantic subspaces in today's data and the need for document learning within these subspaces. We also look at an existing method of incorporating category information in the document vector and present our proposed modification to extend this format to include a category hierarchy. We also present an intuitive method of subspace detection based on the semantic separation of data. We further introduce a general classifier independent semantic subspace learning architecture which can be implemented with any base classifier.

3.1.1 Semantic Subspaces

A vast data repository such as the web contains many broad domains of data which are quite distinct from each other e.g. medicine, education, sports and politics. Each of these domains constitutes a subspace of the data within which the documents are similar to each other but quite distinct from the documents in another subspace. The data within these domains is frequently further divided into many subcategories. While searching for a document in a huge data space, it will be very useful to accurately narrow the search at an initial stage. This will speed up the search as well as allow us to focus on small differences between similar documents. Subspace Learning is therefore receiving increased attention nowadays. Non-overlapping subspaces can be represented as first level topics in a tree structured category hierarchy. In the following sections, we develop a method to encode the relevant subspace within a document vector for fast subspace detection and processing.

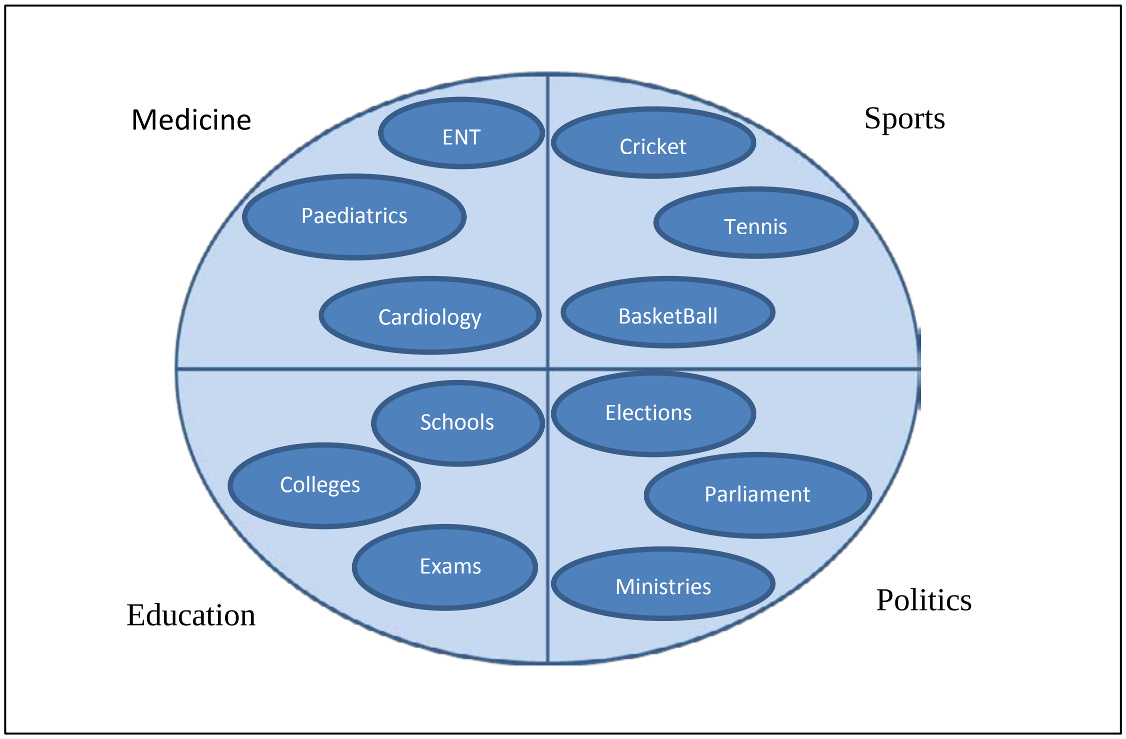


Fig 3.1: Semantic Subspaces in a Large Data Space

3.1.2 Significance Vectors: A Category Based Vector Format

The Significance Vector (Wermter, 1995), (Wermter, Panchev, & Arevian, 1999) is an existing vector representation that incorporates category information. It represents the significance of the data and weighs different words according to their significance for different topics. Significance Vectors are determined based on the frequency of a word in different semantic categories. A modification of the significance vector called the semantic vector uses normalized frequencies where each word *w* is represented with a vector (c_1,c_2,..,c_n) where c_i represents a certain semantic category and *n* is the total number of categories. A value v(w, c_i) is calculated for each element of the semantic vector as the normalized frequency of occurrences of word *w* in semantic category c_i (the normalized category frequency), divided by the sum of the normalized frequency of occurrences of the word *w* for all categories in the corpus

$$v(w, c_i) = \frac{\text{Normalised Frequency of } w \text{ in } c_i}{\sum_k \text{Normalised Frequency of } w \text{ in } c_k}$$
$$\text{where } k \in \{1..n\} \qquad \text{........ } \textit{equation (1)}$$

For each document, the document semantic vector is obtained by summing the semantic vectors for each word in the document and dividing by the total number of words in the document. Henceforth it is simply referred to as the *Significance Vector*. This vector representation was designed for a flat classification system and the positioning of the categories as components of the word/document vector does not follow any specific structure. The following section addresses the need for a structural representation.

3.1.3 Conditional Significance Vectors: A Proposed New Vector Format for Semantic Subspace Learning

The Significance Vectors can be modified to represent a category hierarchy rather than a flat category structure. Consider the two-level hierarchy shown in Fig 3.2 with four level 1 topics (main topics) and 20 level 2 topics (subtopics). Two separate flat significance vectors will be generated for level 1 and level 2. The level 1 significance vectors will consist of 4 vector components representing the four level 1 categories whereas the level 2 significance vector will consist of 20 vector components representing the 20 level 2 categories. Concatenating these two vectors will give a combined vector with 24 components out of which the first 4 represent the 4 level 1 (main) topics and the remaining 20 represent the 20 level 2 (sub) topics. The four level 1 topics can be considered as representing four subspaces of the full data space. Within the 20 level 2 topics, the subtopics belong to the same main topic can be positioned consecutively in the vector space. This will lead to a semantic division of the vector space into 4 groups, each group representing the subtopics of a specific main topic and therefore a *subspace*.

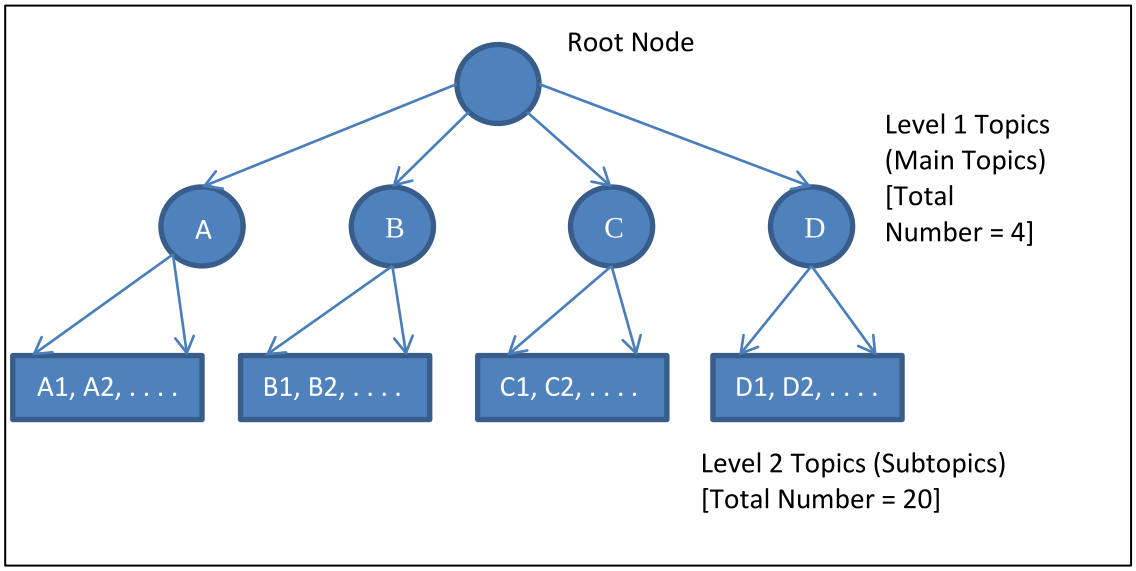

Root Node
Level 1 Topics (Main Topics) [Total Number = 4]
A
B
C
D
A1, A2, ...
B1, B2, ...
C1, C2, ...
D1, D2, ...
Level 2 Topics (Subtopics) [Total Number = 20]

Fig 3.2: An Example Two Level Category Hierarchy

Since the document significance vector represents the significance of the document for the different categories, the category with the maximum numerical significance value is *most likely* to be real the category of a given document. Hence we propose the *Maximum Significance Value* as a means to detect the relevant subspace (level 1 topic) of a new test document.

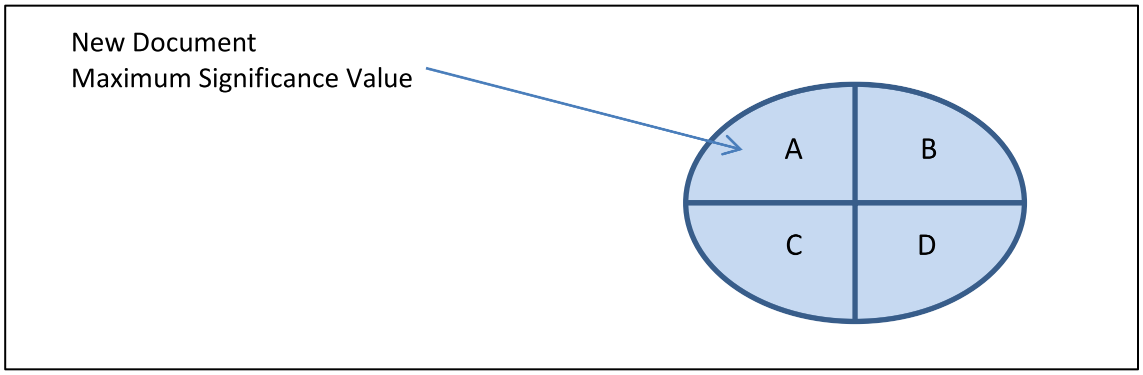
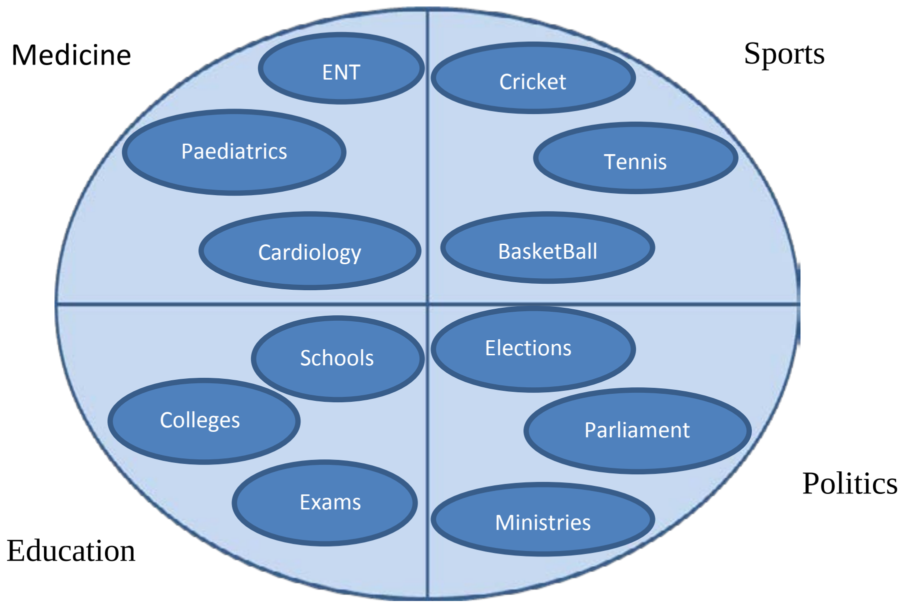


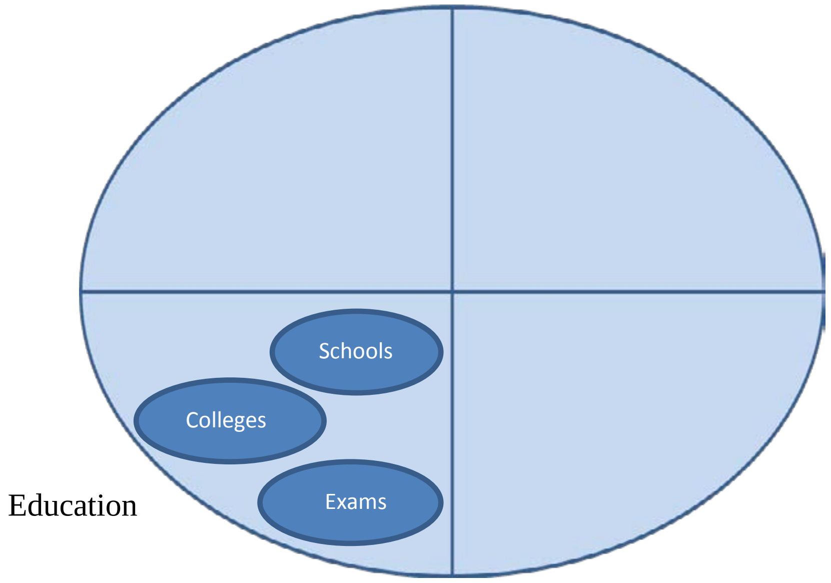
Fig 3.3: Subspace Detection with Maximum Significance Value

Here we first look at the significance vector entries of the first four components of the document vector. These represent the four main topics or the four subspaces of our example. The maximum numerical value among the four level 1 category vector entries is then designated as the Maximum Significance Value. The level 1 category corresponding to this Maximum Significance Value is then most likely to be the main category of the given document. This category is then nominated the level 1 topic or subspace corresponding to that document. Once the subspace has been detected we need to consider only the subtopics present in that subspace for further classification. The subtopics belonging to the other subspaces can be removed from consideration by masking them i.e. setting their values to zero.

The word significance vector values for the level 2 topics will normally be generated by considering all the 20 subtopics in equation (1) i.e. the significance is considered across the whole data space. Therefore we designate this vector as the *Full Significance Word Vector.* However, the importance or significance of a word depends on the subspace in which it occurs e.g. the word “bank” will have a much higher significance in the financial sector (Bank of England) than in the sports sector (bank on a player). Therefor we propose that in the generation of significance values for a word according to equation (1), its occurrence only within the subtopics of a given main topic should be considered. Since this will reduce the value in the denominator of equation (1), the numerical significance value will increase. This will help in distinguishing between the subtopics of a given main topic. We designate this word vector as the *Conditional Significance Word Vector.* The document vectors generated using these two different word vectors will now be called the *Full Significance Document Vector* and the *Conditional Significance Document Vector* respectively.



Full Significance considers the occurrence of a word in all subtopics of all subspaces (main topics)



Conditional Significance considers the occurrence of a word in the subtopics of only a particular subspace (main topic)

Fig 3.4: The Concept of Full & Conditional Significance

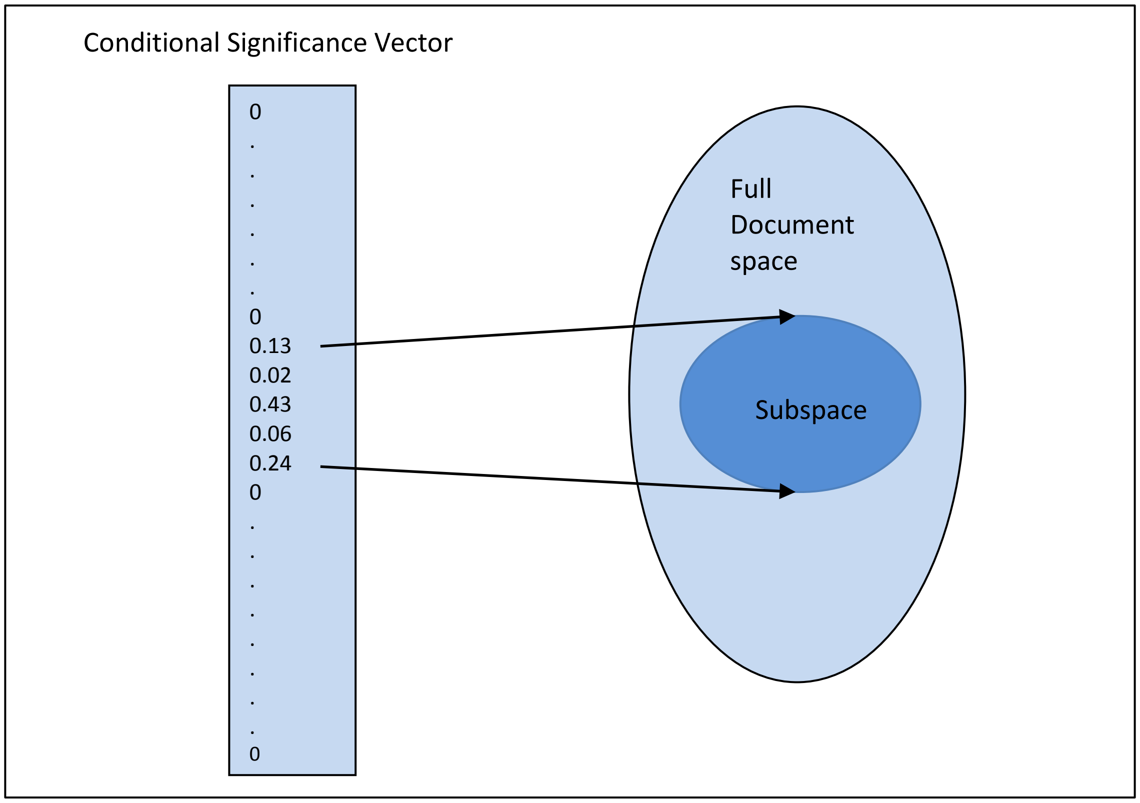


Fig 3.5: Mapping of Conditional Significance Vector to relevant subspace

<u>Example of Conditional Significance Vector Generation</u>

a) Word Conditional Significance Vector

Let us take the frequency of occurrence of a word ***w*** in the main topics and subtopics of the example category hierarchy in Fig. 3.2 as given in the following tables. For simplicity, we take non-normalised frequencies here *(In actual experiments, normalised word frequencies are taken to accommodate different sized documents and different number of documents in each category)*. The corresponding semantic vector values are calculated according to equation (1) using the given non-normalised frequencies. Each of the tables is treated as independent for the purpose of semantic value calculation. The calculated semantic values are also shown in each table.

Main Topics:

	A	**B**	**C**	**D**	**Total Word Freq**
Word Frequency	58	7	17	2	84
Semantic Vector	58/84 = 0.690	7/84 = 0.083	17/84 = 0.202	2/84 = 0.024	

Subtopics of main topic A

	A1	**A2**	**A3**	**A4**	**A5**	**A6**	**A7**	**A8**
Word Freq	4	8	6	10	12	6	4	8
Semantic Vector	0.069	0.138	0.103	0.172	0.207	0.103	0.069	0.138

Subtopics of main topic B

	B1	**B2**	**B3**	**B4**
Word Freq	1	0	2	4
Semantic Vector	0.143	0	0.286	0.571

Subtopics of main topic C

	C1	**C2**	**C3**	**C4**	**C5**	**C6**
Word Freq	4	6	2	1	0	4
Semantic Vector	0.235	0.353	0.118	0.059	0	0.235

Subtopics of main topic D

	D1	**D1**
Word Freq	1	1
Semantic Vector	0.5	0.5

The main topic table gives the Level 1 semantic word vector while the concatenation of the subtopic tables gives the Level 2 semantic word vector as follows:

Level 1:	0.690	0.083	0.202	0.024						
Level 2:	0.069	0.138	0.103	0.172	0.207	0.103	0.069	0.138	0.143	0
	0.286	0.571	0.235	0.353	0.118	0.059	0	0.235	0.5	0.5

The Level 1 and Level 2 semantic vectors are then further concatenated to give the final 24-component word conditional significance vector as follows:

0.690	0.083	0.202	0.024	0.069	0.138	0.103	0.172	0.207	0.103
0.069	0.138	0.143	0	0.286	0.571	0.235	0.353	0.118	0.059
0	0.235	0.5	0.5						

b) Document Conditional Significance Vector

The document significance vector is obtained by summing the significance vectors of all the words present in the document and then dividing by the total number of words in that document. Taking the very simplistic case of a document consisting of only one word ***w,*** the document significance vector is the same as the word conditional significance vector of word ***w*** above. As explained in section 3.1.3, the maximum significance value entry in the level 1 vector defines the main topic which in this case is topic **A**. The level 2 entries belonging to other main topics (**B, C** and **D**) are set to zero. The first 4 vector entries denoting the 4 main topics are then deleted leaving a document vector with 20 components as follows:

0.069	0.138	0.103	0.172	0.207	0.103	0.069	0.138	0	0
0	0	0	0	0	0	0	0	0	0

This vector is the final document conditional significance vector.

3.1.4 Semantic Subspace Learning Architecture

Fig 3.6 shows a general classifier independent subspace learning framework. In this framework, a text document is converted to a two-level significance vector format using the pre-generated conditional significance word vectors. Subspace detection is done based on the *Maximum Significance Value* and the Conditional Significance Vectors are generated as described in the previous section.

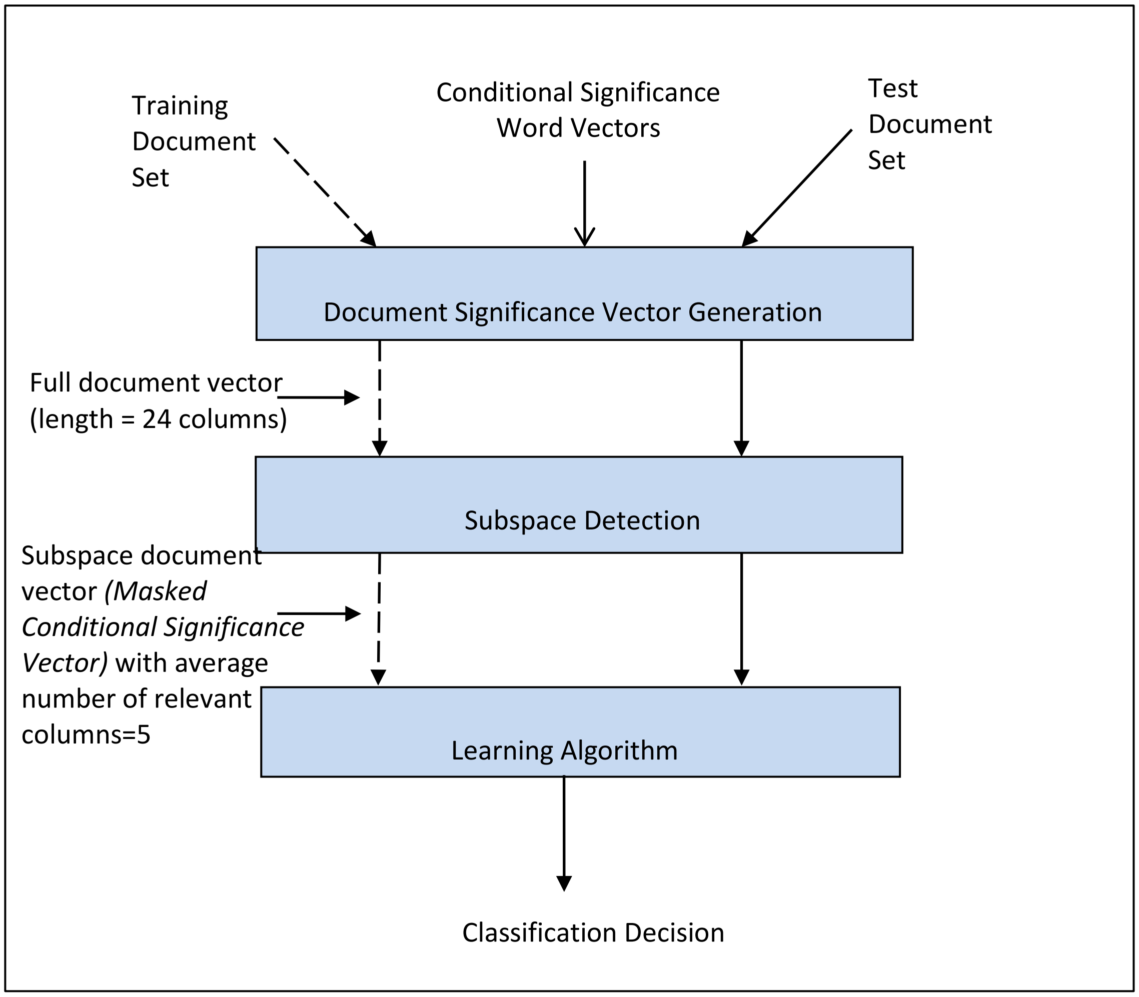

Training Document Set
Conditional Significance Word Vectors
Test Document Set
Document Significance Vector Generation
Full document vector (length = 24 columns)
Subspace Detection
Subspace document vector *(Masked Conditional Significance Vector)* with average number of relevant columns=5
Learning Algorithm
Classification Decision

Fig 3.6: Semantic Subspace Learning Architecture

The training document vectors generated this way along with their actual labels are then used to train a base classifier. A test document then goes through the same procedure for the generation of its conditional significance vector. This vector is then presented to the base classifier for category classification. This framework can be used to compare the performances of various classifiers by using them as base classifiers in the *learning algorithm* block. The vector lengths mentioned in Fig 3.6 correspond to the example two level category hierarchy shown in Fig 3.2.

3.1.5 Test Data Corpora

We tested our two proposed architectures (semantic subspace learning architecture and the hybrid parallel classifier architecture) with two different corpora each – the popular Reuters RCV1 benchmark and the LSHTC dataset drawn from the Open Directory Project (ODP) for the ECIR 2010 challenge. The experiments were conducted in WEKA - an open source machine learning environment.

i) <u>Reuters Corpus (RCV1):</u>

The Reuters Corpus (Rose, Stevenson, & Whitehead, 2002) is a well-known test bench for text classification experiments. We used the Reuters Corpus Volume 1 (RCV1) which is a collection of 806,791 news items written by Reuters journalists in 1996 - 1997. The news items are presented in an XML format which later evolved into the NewsML format used by the International Press Telecommunications Council (IPTC). All the news items are tagged with category codes for three separate schemes - topic, region and industry sector. For our experiments, we used the topic classification scheme. For topics, the RCV1 corpus has a hierarchical organization with four major groups. This scheme is well suited to test the classification performance of subspace based architectures. We used the Reuters Corpus headlines as the main dataset for our experiments as they provide a concise summary of each news article. Each Reuters headline consists of one line of text with about 3 – 12 words. Some example Reuters headlines are given below:

"Healthcare Imaging Q2 loss vs profit."

"Questar signs pact to buy oil, gas reserves."

"Ugandan rebels abduct 300 civilians, army says."

"Estonian president faces reelection challenge."

"Guatemalan sides to sign truce in Norway report."

"CRICKET-Australia beat Zimbabwe by 125 runs in one-day match."

"PRESALE - Akron, Ohio."

The topic codes in the Reuters Corpus are organized into four hierarchical groups. These groups have the following main (top level) categories: Corporate/Industrial (CCAT), Economics (ECAT), Government/Social (GCAT) and Markets (MCAT). These four groups each have a hierarchy of codes and the length of the code represents a subcategory’s depth. As a representative test, ten thousand headlines along with their topic codes were extracted from the Reuters Corpus. These headlines were chosen so that there was no overlap at the first level classification. Each headline belonged to only one level 1 category. According to the Reuters Classification scheme, the second level categories are defined as C1, C2, E1, M1, etc. (code length = 2). However, a study of the Reuters tagged news items shows that these codes are never used in practice. This fact is also confirmed by Rose et al. (2002). To define the subcategory of a news item, the Reuters news items have been tagged with codes such as C12, E21, M11, etc. which are actually third level categories with code length equal to 3. We therefore considered these topics as the direct subtopics of CCAT, ECAT and MCAT. While the subtopics of CCAT, ECAT and MCAT follow a coherent alphanumeric scheme, the sub-categorisation of GCAT (Government/Social) seems to have been done in a more ad-hoc manner. Most of the subcategories of GCAT have abbreviations like GDEF (Defence), GEDU (Education), GSPO (Sports), etc. Hence we took these abbreviations as the direct subtopics of GCAT.

Since most headlines had multiple level 2 subtopic categorisations, the first subtopic was taken as the assigned subtopic. Our assumption here was that the first subtopic used to tag a particular news item was the one most relevant to it. Thus each headline had two labels associated with it – the main topic (Level 1) label and the subtopic (Level 2) label. A total of 50 subtopics were used in our dataset. Headlines were then pre-processed to separate hyphenated words to avoid such combinations being interpreted as new words rather than a sequence of known words. Dictionaries with term frequencies were generated using the TMG toolbox (Zeimpekis & Gallopoulos, 2005) and were then used to generate the Full Significance Document Vector and the Conditional Significance Document Vector (see section 3.1.3) and the tf-idf (Manning, Raghavan, & Schutze, 2008) representation for each document. The Reuters main topics and their distribution in the data along with the number of subtopics of each main topic in our data set are given in Table 3.1. Some of these subtopics along with their numbers present in the data are given in Table 3.2.

We also extracted the full text (headlines + body text) of ten thousand Reuters items and processed them as above to compare the performance of Reuters Full Text with that of Reuters Headlines for the purpose of news classification.

Table 3.1: Reuters Level 1 (Main) Topics

No.	**Main Topic**	**Description**	**Number Present**	**No. of Subtopics**
1.	CCAT	Corporate/ Industrial	4600	18
2.	ECAT	Economics	900	8
3.	GCAT	Government/ Social	1900	20
4.	MCAT	Markets	2600	4

Table 3.2: Some Reuters Level 2 Subtopics

Main Topic	**Subtopic**	**Description**	**Number Present**
CCAT	C17	Funding / Capital	377
CCAT	C32	Advertising/ Promotion	10
ECAT	E12	Monetary/ Economic	107
ECAT	E21	Government Finance	377
GCAT	G15	European Community	38
GCAT	GENV	Environment	30
MCAT	M11	Equity Markets	617
MCAT	M14	Commodity Markets	1050

ii) ***<u>LSHTC Corpus:</u>***

For comparative analysis with data drawn from the web, we used the Large Scale Hierarchical Text Classification (LSHTC) (Kosmopoulos et al., (2010)) competition data from the LSHTC website (http://lshtc.iit.demokritos.gr) as our second corpus. This challenge was part of the European Conference on Information Retrieval (ECIR) 2010. The LSHTC data was constructed by crawling the web pages that are found in the Open Directory Project (ODP) located at www.dmoz.org and translating them into feature vectors. These vectors were called content vectors. The Open Directory Project is an open source and extensive directory of web content. An example web page content accessed from this directory is given below:

"Ambienti Italia brings you world class Italian furniture through infinite selections for decorating your home. Flexibility and design expertise allow us to adapt to any kind of space according to required functions and available dimensions. We want our customers to go home and find the best - comfort and style. Ambienti Italia's collections reflect the achievements and history of Italian home furnishings"

The ODP descriptions of the web pages and the categories were also translated into feature vectors. These vectors were called web page and category description vectors. Two datasets were put up for the LSHTC competition – the large_lshtc_dataset and the smaller dry-run_lshtc_dataset. The directory of each dataset consisted of a *cat_hier.txt* file describing the category hierarchy of the dataset and data folders for four tasks (Task1 – Task4). Task1 contained only crawl data while the data for task 2, task 3 and task 4 contained crawl data and RDF data.

We used the data from the dry-run task1 training folder as our LSHTC corpus. The average number of words in each document in this dataset is 290. This number takes into account only the stemmed words without the stop words. The data is in the form of content vectors which are obtained by directly indexing the web pages. A text file describing the category hierarchy is also given with the data. There were 4463 content vectors in this data file with their associated lowest level labels. We pre-processed these vectors in order to replace the lowest level labels with the corresponding labels of the first two levels of the category hierarchy. We detected 10 level 1 main topics and 158 level 2 subtopics in this dataset. There were no overlapping topics at any level in this corpus. These topics were coded numerically. We replaced this numeric code with an alphanumeric code for ease of analysis. Subsequently the 10 top level categories were given letter codes A – J. These main topics and their distribution in the data along with the number of subtopics of each main topic in this data set are given in Table 3.3. The subtopics were coded A01-A19, B01-B36, etc with the first character denoting the main topic to which these subtopics belonged. The number of document content vectors for some of these subtopics is given in Table 3.4. These vectors were then used to generate the Full Significance Document Vector, the Conditional Significance Document Vector and the tf-idf representation for each document.

Table 3.3: LSHTC Level 1 (Main) Topics

No.	Main Topic	Number Present	Number of Subtopics
1.	A	802	19
2.	B	979	36
3.	C	639	17
4.	D	269	17
5.	E	158	5
6.	F	20	3
7.	G	578	19
8.	H	364	6
9.	I	321	14
10.	J	333	22

Table 3.4: Some LSHTC Level 2 Subtopics

Subtopic	Number Present		Subtopic	Number Present
A09	120		F02	11
A16	8		F03	8
B06	114		G07	47
B26	40		G14	208
C05	2		H02	336
C10	232		H04	2
D02	26		I03	91
D08	62		I10	18
E03	40		J06	44
E05	2		J22	19

3.1.6 The Experimental Environment

Our semantic subspace learning architecture is a general framework which can be implemented with any classifier. To decide which classifiers are best suited to this architecture, we decided to compare the performance of a wide variety of classifiers. WEKA (Hall et al. (2009)), an open source machine learning environment, provided an excellent platform for these experiments as it contains a wide variety of classifiers. WEKA (Waikato Environment for

Knowledge Analysis) was developed by the University of Waikato, New Zealand. We chose ten classifiers in WEKA to cover a wide range of classifier types.

<u>Selected Classification Algorithms:</u>

The ten classification algorithms selected for our experiments were Random Forest, C4.5, Multilayer Perceptron, BayesNet, IBk, NNge, PART, Bagging, LogitBoost and Classification via Regression. Random Forests (Breiman, 2001), (Bernard, Heutte, & Adam, 2009) are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently. C4.5 (Quinlan, 1993), (Ruggieri, 2002) is an inductive tree algorithm with two pruning methods: subtree replacement and subtree raising. The Multilayer Perceptron (Verma, 1997), (Popescu, Balas, Perescu-Popescu, & Mastorakis, 2009) is a neural network which uses backpropagation for training. BayesNet (Pernkopf, 2007), (Likforman-Sulem & Sigelle, 2008) implements Bayes Network learning using various search algorithms and quality measures. IBk (Aha, Kibler, & Albert, 1991) is a k-nearest neighbour classifier which can select an appropriate value of k based on cross-validation and can also do distance weighting. NNge (Martin, 1995) is a nearest neighbor - like algorithm using non-nested generalized exemplars which can be considered as if-then rules. A PART (Frank & Witten, 1998) decision list uses C4.5 decision trees to generate rules. Bagging (Breiman, 1996) is a method for generating multiple versions of a classifier and using these to get an aggregated classifier. LogitBoost (Friedman, Hastie, & Tibshirani, 2000) performs classification using a regression scheme as the base learner. In Classification via Regression (Frank, Wang, Inglis, Holmes, & Witten, 1998), one regression model is built for each class value. Table 3.5 shows the different classification algorithms used with their default parameters in Weka. All these algorithms can cope with categories of different sizes. This takes care of the different number of instances present for each category in the dataset.

Table 3.5: Classification Algorithms and their default parameters in Weka

No.	Algorithm	Parameters
1.	Random Forests	Number of trees = 10; Depth of each tree=unlimited; No. of random attributes = log_2(No. of total attributes) + 1; seed=1
2.	J48 (C4.5)	Confidence factor=0.25, MinNumObj=2, NumFolds=3, Subtree raising = true; seed=1
3.	Bagging	Bag Size Perc=100; Number of Iterations=10; NumExecutionSlots=1; Base Classifier=REP Tree; seed =1
4.	Classification via Regression	Classifier = M5P
5.	LogitBoost	Number of Iterations =10, Number of Runs =1, Shrinkage =1.0,Weight threshold =100, Base Classifier = Decision Stump; seed=1
6.	PART	Confidence factor=0.25, MinNumObj=2, NumFolds=3; seed=1
7.	IBk	KNN=1; No cross validation; No distance weighting; Window size = Unlimited; Uses Linear Nearest Neighbour search algorithm;
8.	BayesNet	Estimator=Simple Estimator, Search algorithm=K2
9.	NNge	NumAttemptsGeneOption=5 NumFoldersMIOption=5
10.	MultiLayer Perceptron	Learning Rate=0.3,Momentum=0.2, Training time=500, Number of Hidden Layers = 1 Number of Hidden Layer units = (attributes + classes) / 2 seed=0

Table 3.6: Variation in seed value used for multiple experimental runs

Chapter of Reported Experiments	Train/Test Split Used on Full Data	Number of Different Seed Values Taken	Variation Range of Seed Value
Chapter 4	50%	10	1 – 5000 (Reuters) 1 – 2232 (LSHTC)
Chapter 5	90%	10	1 – 9000 (Reuters) 1 – 4000 (LSHTC)
Chapter 6	90%	10	1 – 9000 (Reuters) 1 – 4000 (LSHTC)

Table 3.7: Other parameter variations used in Chapter 5 and Chapter 6

Classification Algorithm	Parameters and their different values taken
PART	Confidence factor = 0.25, 0.5, 0.1 Minimum Number of Instances per Leaf (minNumObj) = 2, 3, 4
NNge	Number of attempts for generalization = 2, 4, 5, 7, 9, 10, 12, 15, 17, 20 Number of folders for mutual information = 2, 4, 5, 7, 9, 10, 12, 15, 17, 20
J48	Confidence factor = 0.25, 0.5, 0.1 Minimum Number of Instances per Leaf (minNumObj) = 1, 2, 4
Random Forest	Number of Trees = 10, 15, 20, 25, 30
Multilayer Perceptron	Learning rate = 0.1, 0.3, 0.5, 0.9 Momentum = 0.2, 0.3, 0.5 Number of Hidden layer units (Reuters) = 25, 50, 100 Number of Hidden Layer units (LSHTC) = 79, 158, 316 Training Time = 500, 700, 1000

An average of ten runs was used record the experimental results. Table 3.6 and Table 3.7 show the variation in parameters used for the different runs.

3.1.7 Experimental Methodology

For the experiments, various datasets were generated from the test corpora as explained below:

i) Reuters Headlines Datasets

The initial experiments were run with ten thousand Reuters Headlines. To study the effect of masking out portions of the vector space, datasets for five different vector representations were generated. The Full Significance Vectors were processed in different ways to generate four different data sets. The fifth set was the Conditional Significance Vector dataset.

*a) **No Mask Full Significance Data Set***

For each vector the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were ignored leaving a vector with 50 columns representing 50 subtopics. The order of the data vectors was then randomised and divided into two sets – training set and testing set of 5000 vectors each.

b) Mask 1 Full Significance Data Set

For each document vector the numerical entries in the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were compared. The topic with the minimum numerical value entry was identified. This signified the main topic which was least relevant to the document vector. Therefore the entries for all subtopics belonging to this main topic were masked i.e. set to zero. Finally, the first four columns representing four main categories were deleted. The resultant vector had 50 columns representing 50 subtopics but the subtopic entries for the topic with least significance value had been masked to zero. The average number of relevant columns was then 38. The dataset was then randomised and divided into two sets – training set and testing set of 5000 vectors each.

c) ***Mask 2 Full Significance Data Set***

As above, the numerical entries in the first four columns of each vector representing four main topics CCAT, ECAT, GCAT and MCAT were compared. The main topics with the two smallest numerical value entries were identified. Then the entries for all subtopics belonging to these two main topics were masked i.e. set to zero. Then, the first four columns representing four main categories were ignored. The resultant vector had 50 columns representing 50 subtopics but the subtopic entries for the two topics with the two smallest significance values had been masked to zero. The average number of relevant columns in this case became 25. The masked dataset was then randomised and divided into training and testing sets of 5000 vectors each.

d) ***Mask 3 Full Significance Data Set***

Here again, the numerical entries in the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were compared. The topics with the three smallest numerical value entries were identified. Then the entries for all subtopics belonging to these three topics were masked i.e. set to zero. Finally, the first four columns representing four main categories were deleted. The resultant vector had 50 columns representing 50 subtopics but the subtopic entries for the three main topics with least significance value, 2nd least significance value and 3rd least significance value had been masked to zero. Since there are four main topics in the Reuters corpus, this has the same effect as allowing only the subtopics of the main topic with the maximum significance value in the resultant vector while masking out all the rest. The average number of relevant columns here was 12. Again the dataset was randomised and divided into training set and testing set of 5000 vectors each.

*e) **Mask 3 Conditional Significance Data Set***

In this case, while calculating the significance vector entries for each word in a subtopic, its occurrence in all subtopics *of only a particular main topic* was taken into account. Hence this representation was called the *conditional significance vector*. Here, when calculating significance values for C11, C12, etc, the topics considered were only the subtopics of CCAT. Similarly for M11, M12, etc only MCAT subtopics were considered. For each word, four separate conditional significance sub-vectors were generated for the four main Reuters topics. These sub-vectors were then concatenated together along with the significance value entries for the four main topics to form the 54 column word vector. The Conditional Significance document vector was generated by summing the conditional significance word vectors for each word appearing in the document and then dividing by the total number of words in the document. This vector representation is used to measure the significance of a word within a particular main topic. Hence only the subtopic entries for the main topic with maximum value entry were allowed. All the subtopic entries belonging to the other 3 main topics were masked out. The dataset was then randomised and divided into two sets – training set and testing set of 5000 vectors each. Fig 3.7 shows the Conditional Significance Vector (CSV) and the Mask 3 Full Significance Vector (FSV) for two different Reuters headlines. The main topic label and subtopic label are shown at the end of each vector. As can be seen, the vector entries are boosted in the case of CSV – thus helping to differentiate between subtopics within the subspace.

Headline 1

0……0, 0.20, 0.03, 0.04, 0.02, MCAT/M11 : FSV

0……0, 0.59, 0.11, 0.20, 0.08, MCAT/M11 : CSV

Headline 2

0…..0, 0.03, 0.05, 0.04, 0.0099, 0.01, 0.0073, 0.25, 0.0069, 0…..0, ECAT/E51: FSV

0…..0, 0.13, 0.13, 0.10, 0.0100, 0.02, 0.0300, 0.52, 0.0300, 0…..0, ECAT/E51: CSV

Fig 3.7: Conditional Significance and Full Significance (Mask 3) Vectors for Two Different Reuters Headlines

f) TFIDF Vector Set

The tf-idf weight (Term Frequency–Inverse Document Frequency) is often used in text mining and information retrieval. It is a statistical measure which evaluates how important a word is to a document in a data set. This importance increases with the number of times a word appears in the document but is reduced by the frequency of the word in the data set. Words which occur in almost all documents have very little discriminatory power and hence are given very low weight. The TMG toolbox (Zeimpekis & Gallopoulos, 2005) was used to generate TFIDF vectors for the ten thousand Reuters headlines used in these experiments. Dimensionality reduction was also done using PCA with the same toolbox. The number of dimensions was chosen as 50 for PCA to have vectors similar in size to the significance vectors generated earlier. The TFIDF and the TFIDF/PCA datasets were then randomised and divided into two sets - training and test of 5000 vectors each.

ii) LSHTC Datasets

The web based dataset LSHTC had ten main topics. To study the effect of masking on this dataset, we followed the method explained in the earlier section (for the Reuters dataset) and generated ten LSHTC Full Significance Vector variations – No Mask, Mask 1, Mask 2, Mask 3, Mask 4, Mask 5, Mask 6, Mask 7, Mask 8 and Mask 9. Mask 9 represented the status where only the subtopics of the maximum significance main topic were allowed while the subtopics corresponding to the remaining 9 main topics were masked out. The Mask 9 Conditional Significance Vector and the tf-idf vectors were also generated for this dataset.

iii) Experiment Runs

We ran a wide range of experiments in Weka running each classification algorithm in Table 3.5 for each of the datasets mentioned in (i) and (ii) above. For each experiment, we took ten runs with different seed values and recorded the average classification accuracy and variance of these ten runs. The results of these experiments are discussed in the following chapters.

3.2 Phase II : Hybrid Parallel Classifiers

In this section we propose an extension to the semantic subspace learning architecture presented in section 3.1.4 to deal with very wide variations in data present in today's world (e.g. the web). The previous architecture proposed only one classifier which had to distinguish between all the subtopics at level 2. The vector length to be handled by the classifier also remained the full document vector length even though many components were masked to zero in the conditional significance vector representation.

3.2.1 Hybrid Classifiers and Dimensionality Reduction

Subspace analysis lends itself naturally to the idea of hybrid classifiers. Since each subspace can be viewed as an independent dataset, different classifiers can be used to process different subspaces. Each subspace can be processed by a classifier best suited to the characteristics of that particular subspace. Additionally, instead of using the complete set of full space feature dimensions, classifier performances can be boosted by using only a subset of the dimensions. The use of a smaller number of dimensions will avoid the *curse of dimensionality* and lead to simpler classifiers with lower training times

The method of choosing a feature dimension subset is an area of active research. In the Random Subspace Method (RSM) (Ho, 1998), classifiers were trained on randomly chosen subspaces of the original input space and the outputs of the models were then combined. Several variations of this method have also been proposed in literature [(Garcia-Pedrajas & Ortiz-Boyer, 2008), (Kotsiantis, 2009), (Yaslan & Cataltepe, 2010)]. However random selection of features does not guarantee that the selected inputs have necessary distinguishing information. Today's data contains many distinct semantic subspaces and it will be useful if the selected feature subset was in some way related to the semantic subspace. Our conditional significance vector (sec 3.1.3) proposes that all subtopics of a given main topic be positioned adjacent to each other in the vector space. This vector representation thus divides the vector space into distinct semantic vector component groups. These separate vector component groups can thus be used to train separate classifiers corresponding to the different semantic subspaces.

3.2.2 Hybrid Parallel Classifier Architecture

We propose the use of a parallel classifier combination where different classifiers operate on different portions of the input data space. The combining

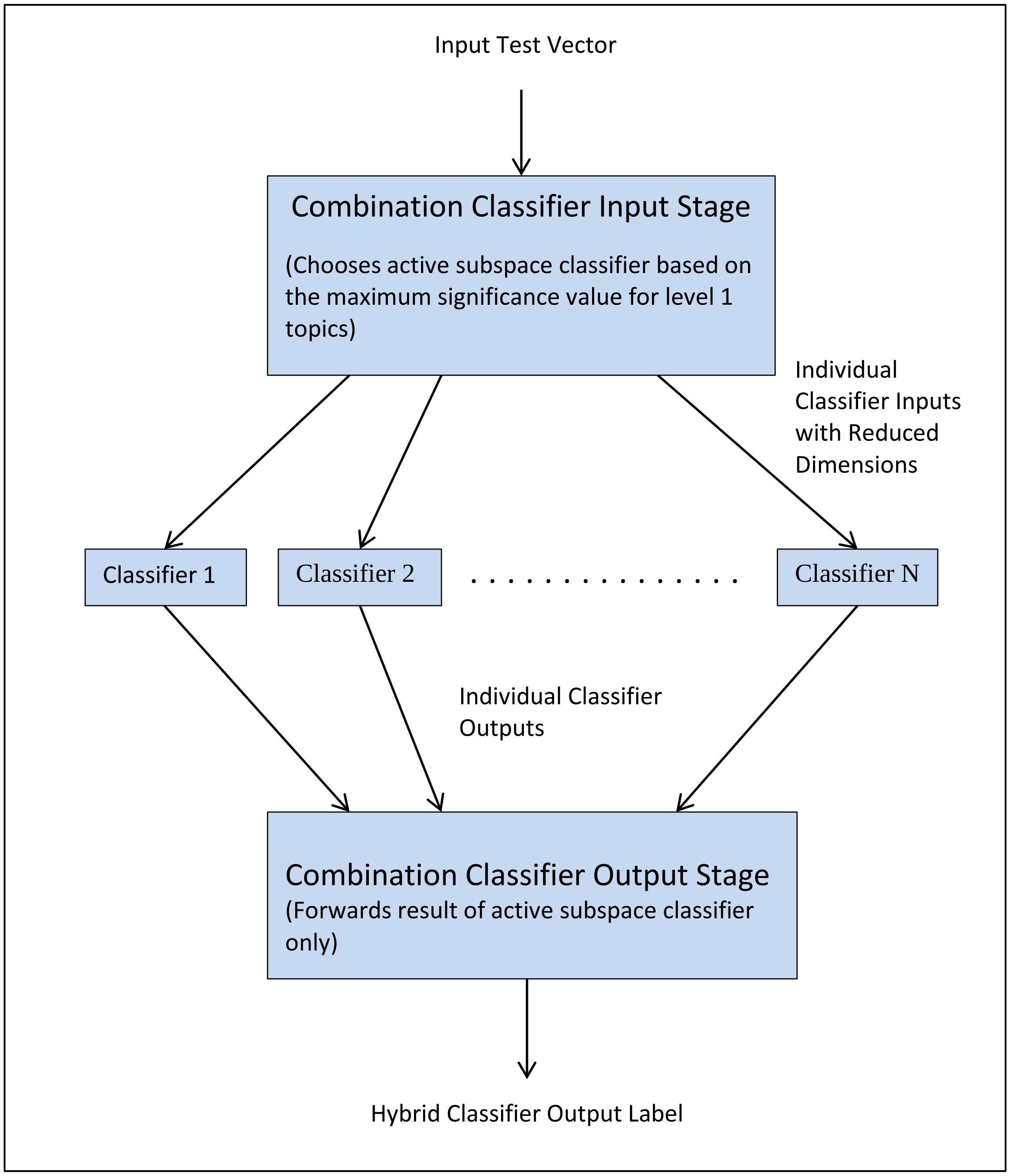

Input Test Vector
Combination Classifier Input Stage
(Chooses active subspace classifier based on the maximum significance value for level 1 topics)
Individual Classifier Inputs with Reduced Dimensions
Classifier 1
Classifier 2
Classifier N
Individual Classifier Outputs
Combination Classifier Output Stage
(Forwards result of active subspace classifier only)
Hybrid Classifier Output Label

Fig 3.8: Hybrid Parallel Classifier Architecture for Semantic Subspace Learning

classifier decides which part of the input data has to be applied to which base classifier. Fig. 3.8 shows our proposed hybrid parallel classifier architecture for semantic subspace learning. During the training phase, the training data set is divided into separate training data subsets according to the level 1 topics or subspaces (4 subsets for our example in Fig 3.2). The relevant feature vector subset is taken for each subspace. These training data subsets with the relevant feature subsets and the associated document subtopic (level 2) labels are then used to train the corresponding base classifiers associated with the different subspaces.

In this architecture the combination classifier chooses the relevant subspace of a test vector based on the *Maximum Significance Value* (see section 3.1.4). Furthermore, only the vector components corresponding to subtopics of this subspace (main topic) are extracted. These relevant vector components are then given to the classifier trained on this subspace for level 2 classification of the test vector. The *predicted* subtopic labels of the test vectors are then compared with their *actual* subtopic labels for the calculation of classification performance.

In this hybrid architecture, each base classifier trains on less data with reduced dimensions. This is expected to reduce the training time of each classifier thus impacting the overall training time. Classification performance is also expected to improve as each base classifier deals with a smaller variation in data.

3.2.3 Experimental Methodology

For these experiments too, we generated as number of datasets from the test corpora as follows:

i) Datasets Generation

As will be described below, three different vector representations (Full Significance Vector, Conditional Significance Vector and tf-idf) were generated for our data. The Conditional Significance Vectors were processed further to generate main category-wise data vector sets (4 different datasets for Reuters and 10 different data sets for LSHTC).

a) *Full Significance Vector*

Here, the document vectors were generated by summing the full significance word vectors for each word occurring in a document and then dividing by the total number of words in that document. For each Reuters Full Significance document vector the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were ignored leaving a vector with 50 columns representing 50 subtopics. The order of the data vectors was then randomised and divided into two sets – training set of 9000 vectors and a test set of 1000 vectors. Similarly, for each LSHTC Full Significance document vector the first ten columns, representing ten main topics (A - J), were ignored leaving a vector with 158 columns representing 158 subtopics. The order of the data vectors was then randomised and divided into two sets – training set of 4000 vectors and a test set of 463 vectors.

b) *Main-Category Wise Conditional Significance Vectors*

Here, the conditional significance word vectors were used to generate the document vectors for the Reuters and LSHTC corpora. These document vectors were then processed as described below to produce the *CSV_RelVectors* for each corpus.

***Reuters Corpus*:** The order of the 10,000 Reuters Conditional Significance document vectors was randomised and divided into two sets – a training set of 9000 vectors and a test set of 1000 vectors. The training set was then divided into 4 sets according to the main topic labels. For each of these sets, only the relevant subtopic vector entries (e.g. C11, C12, etc for CCAT; E11, E12, etc for ECAT) for each main topic were retained. Thus the CCAT category training dataset had 18 columns for 18 subtopics of CCAT. Similarly the ECAT training dataset had 8 columns, the GCAT training dataset had 20 columns and the MCAT training dataset had 4 columns. These 4 training sets were then used to train the 4 parallel classifiers of the Reuters hybrid classifier. The main category of a test data vector was determined by the maximum significance vector entry for the first four columns representing the four main categories. After this, the entries corresponding to the subtopics of this predicted main topic were extracted along with the *actual* subtopic label and given to the classifier trained for this predicted main category.

LSHTC Corpus: The order of the 4463 LSHTC Conditional Significance document vectors was randomised and divided into two sets – training set of 4000 vectors and a test set of 463 vectors. The training set was then divided into 10 sets according to the main topic labels. For each of these for sets, only the relevant subtopic vector entries (e.g. A01, A02, etc for A; B01, B02, etc for B) for each main topic were retained. These 10 training sets were then used to train the 10 parallel classifiers of the LSHTC hybrid classifier. The main category of a test data vector was determined by the maximum significance vector entry for the first ten columns representing the ten main categories. After this, the entries corresponding to the subtopics of this predicted main topic were extracted along with the *actual* subtopic label and given to the classifier trained for this predicted main category.

c) *Main-Category Wise Full Significance Vectors*

To compare the performance of different vector formats, we also generated the main-category wise *Full Significance Vectors*. Here, the Full Significance document vectors were generated as described earlier for the Reuters and LSHTC Corpora. After this, the document vector set for each corpus was divided into main-category wise training and test sets as described in section (b) above. Two variations of the main-category wise Full Significance Vectors were generated for our experiments:

- Main-Category Wise Separated Vectors with the complete set of subtopic vector dimensions (50 for Reuters and 158 for LSHTC) designated as *FSV_FullVector.*

- Main-Category Wise Separated Vectors with only the relevant subtopic vector dimensions corresponding to the actual main category for training vectors and the predicted main category for test vectors. These vectors are designated here as *FSV_RelVector*.

d) *TF-IDF Vector*

The tf-idf vectors were generated as in section 3.1 for both the Reuters as well as the LSHTC Corpus. The tf-idf vector datasets were then randomized and divided into 9000 training vectors / 1000 test vectors for the Reuters Corpus and 4000 training vectors / 463 test vectors for the LSHTC Corpus.

ii) Experiment Runs

We conducted a wide range of experiments in Weka with a large variety of hybrid classifier combinations. We determined the best classifier for each subspace and then combined all these classifiers in a hybrid combination. We also experimented with various predefined two-classifier and four-classifier combinations. Single classifiers using the Full Significance Vector and the tf-idf vector on the complete dataset were used as baselines for these experiments. Classification accuracy and Training Time were used to compare the performances of the various hybrid classifiers with the baselines. The average of ten runs with different classifier parameter values was taken for each recorded performance metric. In some of the experiments the Reuters Full Text document vectors were used to compare the performance of Reuters Headlines v/s Reuters Full Text. We also experimented with parallel classifiers using the same type of classifier for each subspace. The results of all these experiments are discussed in detail in the following chapters.

3.3 Performance Evaluation Metrics and Hypothesis Testing

In a binary classification setting, precision, recall and F-measure are the most popular performance evaluation metrics. However their definition in terms of True Positives, False Positives, True Negatives and False Negatives is not directly applicable to single label multi-way classification. Their extension to single label setting is many times artificially created by considering the single label classification as multiple one-against-rest binary classifications. While this may be required for inherently binary classifiers such as the SVM, it is not at all suitable and even required for direct single label classifiers such as nearest neighbour, tree-based, rule-based, etc. Some researchers (Koller & Sahami , 1997), (McCallum, Rosenfeld, Mitchell, & Ng, 1998) are therefore bypassing these metrics and directly measuring Classification Accuracy which is percentage of the correctly classified instances. Sebastiani (2005) states the evaluation measures used for single label classification are Classification Accuracy (the percentage of correct classifications) and Error which is the converse of Accuracy (Error = 1 - Accuracy). Fukumoto & Suzuki (2002) further claim that in the single label case, classification accuracy is equivalent to standard precision and standard recall. Hence we take Classification Accuracy as a measure of effectiveness for evaluating our architecture. We also measure the timing efficiency of the Hybrid Parallel Classifier for training as well classification/test.

Our main hypothesis is that the use of separate classifiers for separate subspaces along with the use of Conditional Significance Vectors will improve overall subspace classification accuracy and learning time. Hence the main experiments that we conduct are the Conditional Significance Vector experiments in Phase I and the Hybrid Parallel Classifier experiments in Phase II using Reuters Headlines as well as the LSHTC datasets. For comparison, we also conduct the baseline experiments using single classifiers for the complete datasets with two different vector formats – the Full Significance Vector and the standard tf-idf vector. We also conduct some experiments using Reuters Full Text to compare its performance with that of Reuters Headlines for the purpose of news classification. For hypothesis testing, we conduct statistical significance tests on the evaluation metrics (classification accuracy, training time and classification time) measured from these experiments. We use the Friedman test for the Reuters Headlines and the LSHTC experiments as there are two baselines for comparison and this test can compare three sets of values. To compare the performance of Reuters Headlines v/s Reuters Full Text, we use the Wilcoxon Signed Rank test as this test compares two sets of values.

3.4 Chapter Summary

In this chapter, we looked at the presence of semantic subspaces in today's data and introduced a category-hierarchy based document vector representation. We developed the concept of *Maximum Significance Value* which is proposed to detect the level 1 category or relevant subspace of a document and also the concept of *Conditional Significance Vectors* which is based on the importance of a word within a particular level 1 category instead of the whole dataset. We proposed a general semantic subspace learning framework using these two concepts. This framework can be implemented with any base classifier to learn the level 2 categories or subtopics within a dataset. We discussed in detail the two corpora (Reuters and LSHTC) used in our experiments. The experimental methodology for semantic subspace learning along with pre-processing steps and the baselines to be used was also presented.

To deal with the very wide variation present in data today (e.g. web data), we expanded the above framework to generate the Hybrid Parallel Classifier architecture which includes separate classifiers for separate subspaces. This classifier also uses the *Maximum Significance Value* and the *Conditional Significance Vectors.* The experimental methodology for this architecture was presented along with the pre-processing steps and the baselines. The machine learning environment used for our experiments was also discussed along with the different classifiers which were proposed for comparison as base classifiers. We showed how the classification accuracy is equivalent to standard precision and standard recall in a single label setting thus obviating the need to measure multiple metrics. We also presented the statistical significance tests used for testing our main research hypothesis.

In the next chapter we describe the actual experiments conducted for testing the Conditional Significance Vectors using the initial semantic subspace learning architecture presented in Phase I (section 3.1) of this chapter. We also discuss in detail the results of these experiments.

Chapter 4

Conditional Significance Vectors: Experiments & Results

In this chapter, we present the results of the experiments conducted to check the effectiveness of the Conditional Significance Vectors using the initial Semantic Subspace Learning architecture. The first phase of experiments was conducted on a subset of the Reuters RCV1 Corpus consisting of 4 main topics and 50 subtopics. Three different sets of experiments were conducted in this phase. The first set of experiments was conducted using 10,000 Reuters Headlines. This was scaled up to 100,000 headlines in the second set of experiments. The third set of experiments used 10,000 Reuters full text items which contained both headlines as well as body text. These three sets of experiments indicated that the use of Conditional Significance Vectors increases classification accuracy. In order to scale up the number of topics, we conducted the second phase of experiments with the LSHTC Corpus which consisted of 10 main topics and 158 subtopics. All the experiments used a train/test split of 50%.

4.1 Upper Limit on Subspace Classification Accuracy using Conditional Significance Vectors

The detection of the relevant subspace of a test document is based on the Maximum Significance Value. We measured the accuracy of choosing the level 1 (main) topic of a document with this method and found it to be 96.80% for Reuters Headlines, 82.50% for Reuters Full Text and 85.31% for LSHTC. These values form the upper limit on subtopic (Level 2) classification accuracy for the corresponding datasets. This is because a wrong subspace selection will propagate this error down to level two and cause a wrong selection of level 2 subtopics as well.

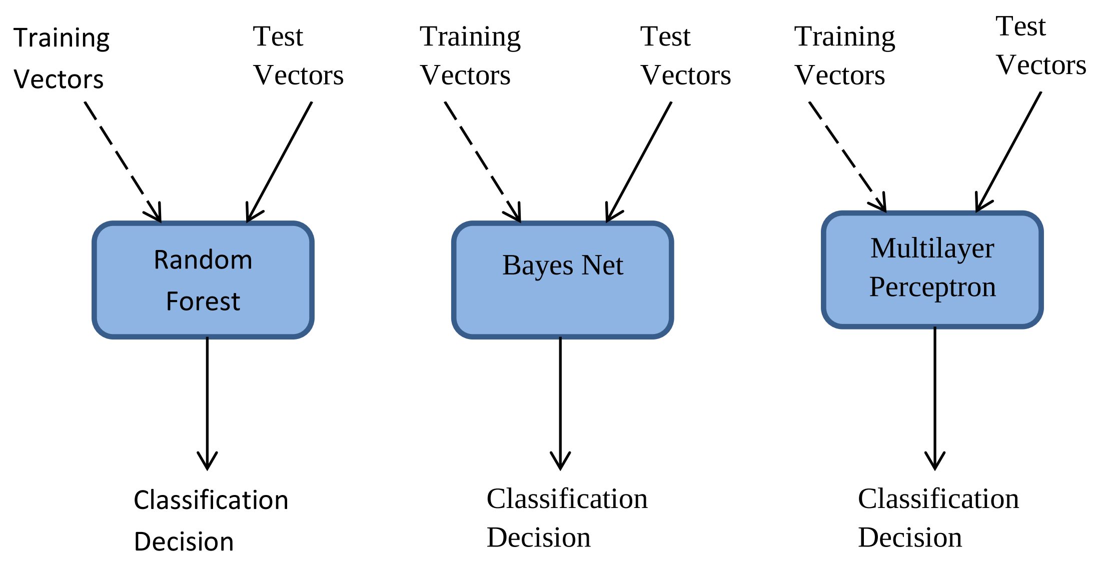
4.2 Experimental Setup for the Initial Semantic Subspace Learning Architecture

The experiments for this chapter were set up as follows:

- Text Dataset converted into Conditional Significance Vectors
- Train/Test split of 50% taken for each vector dataset
- A single classifier trained and tested with this dataset
- Main topic and Subtopic Classification Accuracy recorded for each experiment run
- Average of 10 runs with different parameters used for comparing the performance of different classifiers
- Ten different classifiers compared in this chapter

- Same experiments also run with Full Significance Vectors with different levels of masking for comparison.

The figures below show some examples of the experimental setup for subspace learning. All the vectors shown are the Conditional Significance Vectors.



4.3 Reuters Corpus – Experiments & Results

4.3.1 Case I: 10,000 Reuters News Headlines

Here 10,000 Reuters Headlines were extracted along with their main topic and subtopic labels. This dataset was pre-processed to separate hyphenated words. We generated the category-hierarchy based significance vector representation of this Reuters dataset where the first four vector elements represented the 4 main topics and the remaining 50 vector elements represented the subtopics. We generated both the Full Significance as well as the Conditional Significance versions of the document vector.

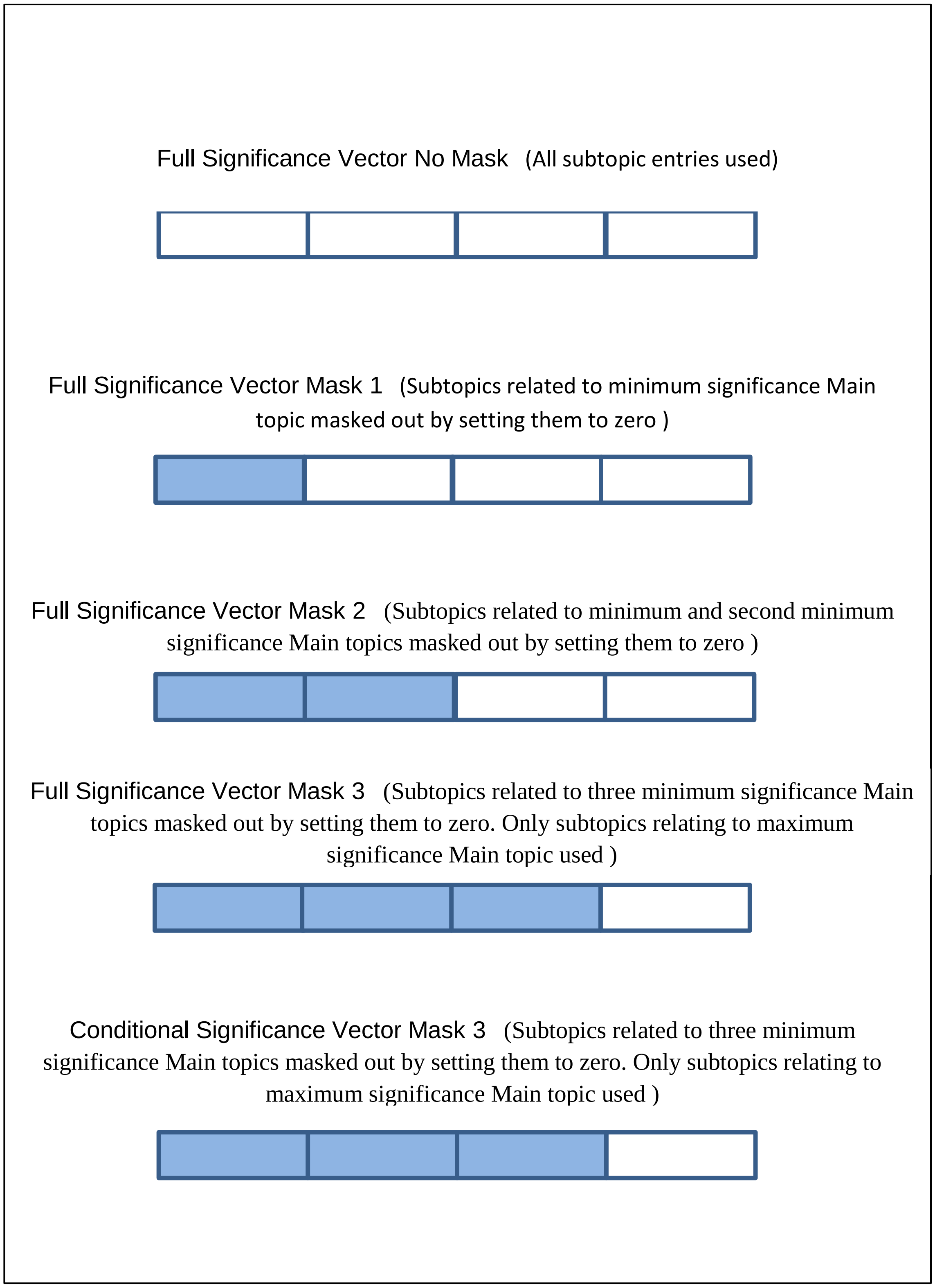

Full Significance Vector No Mask (All subtopic entries used)
Full Significance Vector Mask 1 (Subtopics related to minimum significance Main topic masked out by setting them to zero)
Full Significance Vector Mask 2 (Subtopics related to minimum and second minimum significance Main topics masked out by setting them to zero)
Full Significance Vector Mask 3 (Subtopics related to three minimum significance Main topics masked out by setting them to zero. Only subtopics relating to maximum significance Main topic used)
Conditional Significance Vector Mask 3 (Subtopics related to three minimum significance Main topics masked out by setting them to zero. Only subtopics relating to maximum significance Main topic used)

Fig 4.1: Reuters Corpus Document Vectors with different levels of masking

As described in section 3.1.7, the Full Significance Vector was examined to determine the minimum value among the first four main topic entries. The main topic corresponding to this minimum value was considered the least significant main topic for the given document. The subtopic entries corresponding to this main topic were masked i.e. set to zero. The first four main topic entries were then removed leaving a vector length of 50. This generated the FSV Mask 1 dataset of the Reuters Corpus. Similarly, masking out the subtopic entries of the two minimum value main topics generated the FSV Mask 2 dataset. In a similar fashion, the FSV Mask 3 dataset was generated. FSV Mask 3 represented the maximum masking dataset. We also generated CSV Mask 3 which was the maximum masking dataset using the Conditional Significance document vectors. Fig 4.1 shows the Reuters Corpus Document Vectors with different levels of masking.

Two sets of experiments were run to test learning at the first two levels of Reuters topic classification. Ten runs of each algorithm with different seed values (wherever applicable) were taken for each vector representation. Four algorithms (Classification via Regression, IBk, BayesNet and NNge) did not have the option for entering a random seed value in Weka. Three algorithms (C4.5, LogitBoost and PART) had an option for entering random seed value but the results for all 10 runs were identical. Only three algorithms (Random Forest, Bagging and Multilayer Perceptron) showed variance in the classification accuracy values. The average and variance of the classification accuracy for 10 runs with different seed values was calculated for each algorithm. The results of these experiments are shown in Table 4.1 and Table 4.2 below. The abbreviated column names represent the following vectors:

FS_0: Full Significance Vector No Mask
FS_1: Full Significance Vector Mask 1
FS_2: Full Significance Vector Mask 2

FS_3: Full Significance Vector Mask 3

CS_3: Conditional Significance Vector Mask 3

In Table 4.1, all algorithms except the nearest neighbor algorithm IBk (No. 7) show that the maximum masking option (Mask 3) gives the best result. This indicates that the maximum significance value is a good indicator of the relevant subspace. The best results are divided between FS_3 and CS_3 for different algorithms. In Table 4.2, the Conditional Significance Vector representation with maximum masking option (Mask 3) gives the best average accuracy result with all algorithms. As the Mask 3 option allows only the subtopics of the main topic with maximum Significance Value, This shows that branching on maximum significance value along with the use of conditional significance within a subspace greatly improves classification at level 2.

Table 4.1: Main Topic Average Classification Accuracy for Test Vectors (Reuters 10,000 Headlines)

S. No	**Algorithm**	**FS_0**	**FS_1**	**FS_2**	**FS_3**	**CS_3**	**TFIDF /PCA**
1.	Random Forest	91.17	90.67	91.45	**96.45**	**96.45**	79.46
2.	J48 (C4.5)	92.46	91.02	92.40	95.72	**96.10**	73.58
3.	Bagging	92.24	91.95	93.54	**96.39**	96.29	78.89
4.	Classification Via Regression	92.10	94.94	94.72	96.28	**96.78**	77.54
5.	LogitBoost	92.30	92.22	90.96	96.24	**96.38**	72.20
6.	PART	93.46	92.86	92.20	**95.92**	95.60	74.14
7.	IBk	**96.84**	96.74	95.28	95.44	95.94	76.74
8.	BayesNet	83.58	81.26	71.70	96.26	**96.30**	59.62
9.	NNge	95.66	95.58	89.92	**96.64**	96.34	73.72
10.	Mulitlayer Perceptron	96.54	96.40	95.31	96.49	**97.43**	79.77

Table 4.2: Subtopic Average Classification Accuracy for Test Vectors (Reuters 10,000 Headlines)

S. No	Algorithm	FS_0	FS_1	FS_2	FS_3	CS_3	TFIDF /PCA
1.	Random Forest	82.11	80.69	74.69	88.55	**90.60**	57.37
2.	J48 (C4.5)	87.90	87.62	78.50	88.90	**90.42**	49.16
3.	Bagging	86.68	87.04	79.51	89.53	**92.06**	57.51
4.	Classification Via Regression	92.12	91.94	83.32	91.36	**92.98**	56.02
5.	LogitBoost	92.32	92.10	83.88	91.16	**92.62**	52.98
6.	PART	87.18	87.98	77.20	88.78	**90.24**	50.44
7.	IBk	90.84	90.58	81.76	89.66	**91.22**	55.52
8.	BayesNet	68.52	61.98	52.18	86.84	**89.04**	46.74
9.	NNge	91.30	91.16	82.34	90.96	**92.42**	54.82
10.	Mulitlayer Perceptron	91.96	91.86	82.07	91.39	**92.39**	58.84

Tables 4.3 and 4.4 show the variance in classification accuracy across the ten runs with different seed values taken for each dataset and each algorithm. Low variance is an indicator of a stable classification method as the classification accuracy is not dependent on the seed value which can be random. Table 4.3 shows that the lowest variance in level 1(main topic) classification accuracy is obtained by the maximum masking option (Mask 3). The minimum variance here is given by the Conditional Significance Vector Mask 3 (CS_3). Table 4.4 shows that the lowest variance in level 2 (subtopic) accuracy is also given by the maximum masking option (Mask 3). The minimum variance values are split between the Full Significance Vector Mask 3 (FS_3) and the Conditional Significance Vector Mask 3 (CS_3). As the maximum masking option limits the learning process to within a subspace, the classifiers are better able to distinguish between subtopics in this subspace and thus variance in classification accuracy is less.

Table 4.3: Main Topic Classification Accuracy Variance for ten runs (Reuters 10,000 Headlines)

S. No .	Algorithm	FS_0	FS_1	FS_2	FS_3	CS_3	TFIDF/ PCA
1.	Random Forest	0.227	0.236	0.123	0.018	**0.011**	0.120
2.	J48 (C4.5)	0	0	0	0	0	0
3.	Bagging	0.234	0.084	0.042	**0.003**	**0.003**	0.224
4.	Classification Via Regression	0	0	0	0	0	0
5.	LogitBoost	0	0	0	0	0	0
6.	PART	0	0	0	0	0	0
7.	IBk	0	0	0	0	0	0
8.	BayesNet	0	0	0	0	0	0
9.	NNge	0	0	0	0	0	0
10.	Mulitlayer Perceptron	0.109	0.115	0.095	0.062	**0.042**	0.742

Table 4.4: Subtopic Classification Accuracy Variance for ten runs (Reuters 10,000 Headlines)

S.No.	Algorithm	FS_0	FS_1	FS_2	FS_3	CS_3	TFIDF/ PCA
1.	Random Forest	0.545	1.239	0.264	**0.101**	0.146	0.202
2.	J48 (C4.5)	0	0	0	0	0	0
3.	Bagging	0.233	0.241	0.151	0.030	**0.028**	0.147
4.	Classification Via Regression	0	0	0	0	0	0
5.	LogitBoost	0	0	0	0	0	0
6.	PART	0	0	0	0	0	0
7.	IBk	0	0	0	0	0	0
8.	BayesNet	0	0	0	0	0	0
9.	NNge	0	0	0	0	0	0
10.	Mulitlayer Perceptron	0.141	0.163	0.746	**0.026**	0.059	0.320

In Tables 4.1 – 4.4, the Mask 3 option consistently shows the best results (highest classification accuracy and lowest classification accuracy variance) at level 1 and level 2. This shows that the maximum significance value is successful in identifying the relevant subspace (level 1 topic). Since the Conditional Significance Vector with Mask 3 option encodes the subspace within the vector itself, the subtopic accuracy table shows the combined effect of branching at level one and applying the classification algorithms at level 2. Consistent maximum accuracy obtained at level 2 by the conditional significance vector with all the algorithms shows that conditional significance is successful in differentiating between subtopics within a data subspace. Thus our conditional significance vector representation is unique in that it incorporates both subspace branching and subspace learning in the same step.

Table 4.5: Classification Accuracy with TF-IDF/PCA and TFIDF (Reuters 10,000 Headlines)

S. No.	**Algorithm**	**Main Topic**			**Subtopic**	
		TFIDF/ PCA	**TFIDF**		**TFIDF/ PCA**	**TFIDF**
1.	Random Forest	79.46	75.21		57.37	54.81
2.	J48 (C4.5)	73.58	68.66		**49.16**	**53.68**
3.	Bagging	78.89	72.64		57.51	52.18
4.	Classification Via Regression	77.54	46.22		56.02	21.44
5.	LogitBoost	72.20	65.64		52.98	50.56
6.	PART	74.14	68.00		**50.44**	**53.08**
7.	IBk	76.74	73.82		55.52	52.66
8.	BayesNet	**59.62**	**70.32**		**46.74**	**47.68**
9.	NNge	73.72	71.98		54.82	52.00
10.	Mulitlayer Perceptron	79.77	66.31		58.84	32.61

A comparison of the TF-IDF and the TF-IDF/PCA baselines in Table 4.5 shows that the use of PCA dimensionality reduction provides small improvements in the classification performances for most of the algorithms. For level 1 (main topic), the classification accuracy of the Classification Via Regression algorithm is greatly improved and that of the multilayer perceptron is also improved with the use of PCA while the classification accuracy of BayesNet is degraded significantly by its use. For level 2 (subtopic), the use of PCA slightly degrades the classification accuracy of J48 (C4.5), PART and BayesNet while greatly improving the classification accuracy of the Classification Via Regression algorithm and the Multilayer perceptron. The values in red show the cases where the use of PCA has reduced the classification accuracy. All the other algorithms show small improvements with the use of PCA. Hence, overall, the use of PCA does not seem very beneficial in our case as it does not substantially improve the classification accuracy for all algorithms.

4.3.2 Case II: 100,000 Reuters News Headlines

In this case, we scaled up the experiments to include 100,000 Reuters Headlines which were processed to produce the five different datasets as in section 4.1.1. These datasets were FSV No Mask, FSV Mask 1, FSV Mask 2, FSV Mask 3 and CSV Mask 3. A Train/Test split of 50% was taken to generate 50,000 training and 50,000 test vectors. The Multilayer Perceptron and the two rule-based classifiers (PART and NNge) could not complete the classification process within reasonable time and were removed from consideration. The results obtained are given in Table 4.6 and Table 4.7. In the level 1 (main topic) results shown in Table 4.6, the maximum accuracies (except for IBk and Classification via Regression) are again shown with maximum masking (FS_3 and CS_3).

Table 4.6: Main Topic Classification Accuracy for Test Vectors (Reuters 100,000 Headlines)

S. No	Algorithm	FS_0	FS_1	FS_2	FS_3	CS_3
1.	Random Forest	92.78	93.01	94.38	**94.73**	94.35
2.	J48(C4.5)	91.62	91.86	93.58	**94.21**	93.41
3.	Bagging	93.64	93.66	94.75	**94.98**	94.46
4.	Classification Via Regression	94.21	94.40	**94.93**	94.66	94.21
5.	LogitBoost	90.86	90.57	91.44	**93.97**	**93.97**
6.	IBk	**95.37**	95.21	95.03	93.54	92.72
7.	Bayes Net	85.79	83.40	81.09	**93.96**	**93.96**

Table 4.7: Subtopic Classification Accuracy for Test Vectors (Reuters 100,000 Headlines)

S. No	Algorithm	FS_0	FS_1	FS_2	FS_3	CS_3
1.	Random Forest	78.55	78.99	80.10	82.36	**84.28**
2.	J48(C4.5)	79.45	79.27	80.03	81.78	**82.71**
3.	Bagging	82.64	82.80	83.36	84.22	**85.50**
4.	Classification Via Regression	84.76	85.15	85.16	84.55	**85.71**
5.	LogitBoost	85.44	**85.49**	85.22	83.92	85.47
6.	IBk	**82.70**	82.32	82.06	80.50	81.54
7.	Bayes Net	67.69	59.47	53.24	79.02	**80.74**

The level 2 (subtopic) results in Table 4.7 show that tree-based classifiers (J48 and Random Forest), Bagging, Classification Via Regression and BayesNet show the maximum accuracies with Conditional Significance Vectors. The difference in classification accuracy between the unmasked FS_0 vector and the maximum masked CS_3 vector is much greater with J48, Random Forest and BayesNet than with Bagging and Classification Via Regression. The maximum improvement is shown by BayesNet where the accuracy goes from 67.69% with FS_0 to 80.74% with CS_3. As Bayesian classifiers are very fast, this technique will be very useful when dealing with large datasets. The best performance with LogitBoost (FS_2) was very close to the corresponding performance of CS_3 (85.49% v/s 85.47%). The nearest neighbour classifier (IBk), on the other hand, favours the unmasked basic vector format FS_0.

Overall, these results show that maximum masking is a good indicator of a subspace and that the use of conditional significance vectors improves subtopic classification accuracies for the majority of the classifiers.

4.3.3 Case III: 10,000 Reuters Full Text News Items

Here, ten thousand Reuters Full Text (Headlines + Body Text) items were extracted and processed as described in section 4.1.1 generating the corresponding five datasets (FS_0, FS_1, FS_2, FS_3 and CS_3). The results of running these experiments in Weka are given in Table 4.8 and Table 4.9. An examination of the main topic (level 1) results given in Table 4.8 show that the nearest neighbour classifier gives the highest accuracy with no masking (FS_0), one rule-based classifier (NNge) and the meta-classifier (Bagging, Classification Via Regression and LogitBoost) give the highest accuracy with masking half the data i.e. 2 out of 4 main topics (vector FS_2). The tree based classifiers (Random Forest and J48), the Bayesian classifier BayesNet and a rule-based classifier PART give the highest accuracy with maximum masking. In the subtopic results in Table 4.9, IBk and MLP along with two meta-classifiers (Classification Via Regression and LogitBoost) give the highest accuracy with no masking (FS_0) while PART gives the highest accuracy with FS_1. The remaining five classifiers all give the highest accuracies with the Conditional Significance Vector CS_3. These results show that for long documents, the elementary classifier performances are improved more by the use of Conditional Significance Vectors than the more complex classifiers.

Table 4.8: Main Topic Classification Accuracy for Reuters 10,000 Full Text News Items

S. No	Algorithm	FS_0	FS_1	FS_2	FS_3	CS_3
1.	Random Forest	84.96	86.10	86.26	**86.54**	85.84
2.	J48 (C4.5)	83.48	82.20	83.72	**85.72**	84.74
3.	Bagging	86.38	86.12	**87.64**	86.72	86.40
4.	Classification Via Regression	87.94	87.94	**88.40**	87.98	85.86
5.	LogitBoost	85.92	85.36	**86.02**	84.32	85.78
6.	PART	85.56	85.10	85.88	**86.20**	84.46
7.	IBk	**88.66**	88.60	88.60	86.34	85.26
8.	BayesNet	72.02	71.88	75.08	81.78	**81.80**
9.	NNge	87.16	87.40	**87.58**	86.42	86.04
10.	Mulitlayer Perceptron	89.74	**90.48**	89.22	88.94	86.32

Table 4.9: Subtopic Classification Accuracy for Reuters 10,000 Full Text News Items

S. No	Algorithm	FS_0	FS_1	FS_2	FS_3	CS_3
1.	Random Forest	64.84	65.02	67.66	70.24	**71.12**
2.	J48 (C4.5)	66.96	67.80	67.50	68.78	**70.66**
3.	Bagging	70.32	70.70	72.10	72.00	**74.04**
4.	Classification Via Regression	**79.10**	78.60	78.70	74.58	75.52
5.	LogitBoost	**80.06**	79.28	79.00	74.78	75.48
6.	PART	69.04	**70.12**	68.30	69.18	68.46
7.	IBk	**75.06**	74.30	74.68	71.82	72.40
8.	BayesNet	54.44	38.96	40.36	62.52	**62.80**
9.	NNge	72.28	73.46	73.26	72.12	**73.86**
10.	MultiLayer Perceptron	**77.87**	76.91	76.86	74.07	75.05

4.4 LSHTC Corpus – Experiments & Results

In these experiments, we tested the effect of a much larger set of categories on the performance of the Conditional Significance Vector. The LSHTC Corpus had 10 main and 158 subtopic categories as compared to 4 main and 50 subtopics of the Reuters dataset in the previous section.

Here, we generated the category-hierarchy based significance vector representation of the LSHTC dataset where the first ten vector elements represented the ten main topics and the remaining 158 vector elements represented the subtopics of the LSHTC Corpus. We generated both the Full Significance as well as the Conditional Significance versions of the document vector. The Full Significance Vector was examined to determine the minimum value among the first ten main topic entries. The main topic corresponding to this minimum value was considered the least significant main topic for the given document. The subtopic entries corresponding to this main topic were masked i.e. set to zero. The first ten main topic entries were then removed leaving a vector length of 158. This generated the FSV Mask 1 dataset of the LSHTC Corpus. Similarly masking out the subtopic entries of the two minimum value main topics generated the FSV Mask 2 dataset. In a similar fashion, the FSV Mask 3 – FSV Mask 9 datasets were generated. FSV Mask 9 represented the maximum masking dataset. We also generated CSV Mask 9 which was the maximum masking dataset using the Conditional Significance document vectors.

Table 4.10 and Table 4.11 show the results of learning at the top two levels of the LSHTC dataset. FS_0 - FS_9 represent the Full Significance Vectors with various levels of masking while CS_9 represents the Conditional Significance Vector with the maximum number (9 out of 10) of main topics masked out.

Table 4.10: Main Topic Classification Accuracy for Test Vectors (LSHTC)

Table 4.10 – Part 1

Algorithm	**FS_0**	**FS_1**	**FS_2**	**FS_3**	**FS_4**	**FS_5**
Random Forest	63.41	62.64	63.64	66.34	67.70	70.17
J48	69.21	68.44	69.12	70.15	71.45	72.34
Bagging	71.00	69.97	72.70	72.52	75.44	74.81
Classification Via Regression	73.38	74.23	76.20	77.86	79.07	79.20
LogitBoost	81.58	81.98	82.47	81.80	83.19	84.54
PART	71.72	72.52	71.63	73.15	73.87	72.34
IBk	86.78	85.57	86.02	85.88	85.43	86.37
BayesNet	65.71	62.39	61.27	60.06	60.20	59.12
NNge	73.02	74.54	75.93	80.05	81.53	83.10
MLP	35.23	51.41	36.98	48.72	45.90	20.39

Table 4.10 – Part 2

Algorithm	**FS_6**	**FS_7**	**FS_8**	**FS_9**	**CS_9**	**TFIDF**
Random Forest	71.60	74.92	80.04	**87.01**	86.78	45.21
J48	76.38	76.60	78.80	**83.24**	81.44	38.17
Bagging	78.04	79.16	82.07	**88.08**	87.14	49.24
Classification Via Regression	80.14	82.97	86.06	**87.63**	85.48	46.55
LogitBoost	84.72	85.25	85.93	**87.18**	86.33	51.34
PART	76.11	77.36	79.78	**83.86**	82.21	39.25
IBk	86.82	88.21	**88.61**	87.09	86.51	42.03
BayesNet	62.17	66.20	74.09	**86.91**	86.19	53.00
NNge	83.42	85.88	86.28	**86.96**	86.02	45.79
MLP	22.73	48.32	44.55	**84.58**	82.61	44.44

Table 4.11: Subtopic Classification Accuracy for Test Vectors (LSHTC)

Table 4.11 – Part 1

Algorithm	**FS_0**	**FS_1**	**FS_2**	**FS_3**	**FS_4**	**FS_5**
Random Forest	36.20	35.94	36.87	38.21	40.01	41.48
J48	46.62	47.96	50.47	53.12	55.00	55.49
Bagging	54.50	54.73	54.46	56.88	56.16	58.14
ClassViaRegress	87.14	**87.85**	87.27	87.36	85.93	84.31
LogitBoost	88.35	**88.75**	88.75	88.70	88.75	88.44
PART	52.53	53.74	52.71	60.47	58.76	59.48
IBk	**80.95**	79.20	79.34	77.99	77.81	77.95
BayesNet	39.13	38.95	19.99	11.52	11.07	14.88
NNge	65.13	64.37	66.07	66.34	68.35	69.12
MLP	18.69	41.64	22.37	7.58	28.91	29.49

Table 4.11 – Part 2

Algorithm	**FS_6**	**FS_7**	**FS_8**	**FS_9**	**CS_9**	**TFIDF**
Random Forest	43.08	45.84	50.50	59.15	**59.29**	22.92
J48	55.13	61.23	63.87	**72.43**	71.81	17.72
Bagging	59.79	63.02	68.00	72.88	**73.11**	25.89
Classification Via Regression	82.97	83.28	80.01	80.01	80.28	21.96
LogitBoost	87.85	86.87	86.69	82.21	82.07	27.23
PART	61.45	64.63	67.86	**72.39**	71.85	16.38
IBk	78.98	79.83	80.14	77.86	79.83	21.03
BayesNet	19.32	22.59	28.82	57.91	**61.68**	15.13
NNge	71.49	72.30	74.59	**75.53**	73.55	24.60
MLP	31.02	26.85	31.51	21.96	**68.49**	26.96

As can be seen from Table 4.10, the best main topic classification accuracy is obtained by the maximum masking dataset FS_9 with the CS_9 dataset following close behind. This supports our hypothesis that maximum masking identifies the subspace accurately. Table 4.11 shows that, apart from the nearest neighbour classifier IBk and two meta-classifiers Classification Via Regression and LogitBoost, the best subtopic classification accuracies are split between the two maximum masking options FS_9 and CS_9. A reason for this can be that unlike Reuters, the LSHTC dataset has no multiple label assignments at the subtopic level. The unique advantage of the Conditional Significance Vector is that it increases the separation between subtopics. As the LSHTC subtopics are already separated, the FS_9 and CS_9 vectors in this case would be quite similar to each other. These results are quite similar to the level 1 performance with Reuters Ten Thousand Headlines. In that case, the level 1 topics have no overlap and the best results are divided between the two maximum masking options FS_3 and CS_3.

4.5 Conclusion

In these experiments, we explore semantic subspace learning with the overall objective of improving document retrieval in a vast document space. Our experiments in Reuters Case 1 (Ten Thousand Reuters Headlines) show that the maximum significance value has a good potential to identifying the main (Level 1) topic of a document. They also show that modifying the significance vector (conditional significance) to process only the subspace improves learning within the subspace. Thus the combination of branching on maximum significance value along with using conditional significance improves subspace learning. The subspace detection is done by processing a single document vector. This method is independent of the total number of data samples and only compares the level 1 topic entries. The time complexity is thus O(k) where k is the number of level 1 topics. The novelty of our approach is in the vector representation. In the document conditional significance vector generated by the subspace detection step, the subspace is encoded in the vector (the non-zero values represent the subspace). Secondly, the numerical significance values in the word conditional significance vector denote the significance of a particular word for different subtopics *within* that subspace. Since the document vector is a summation of the word vectors, this helps in differentiating between topics within a given subspace (between subtopics of a main topic in case of Reuters Corpus) thus enhancing subspace learning.

The Reuters Case II (One Hundred Thousand Reuters Headlines) experiments suggest that the basic tree-based and Bayesian classifiers benefit more from the use of Conditional Significance Vectors than the more complex meta-classifiers like boosting and classification via regression. The performance of IBk indicates that nearest neighbor classifiers are not suitable for semantic subspace learning. In section 2.1 of the literature review, nearest neighbor classifiers were shown to be unsuitable for large datasets due to their high run-time memory requirements and large training times. Therefore, we have removed IBk from consideration for later experiments.

The Reuters Case III (Ten Thousand Reuters Full Text News Items) experiments further confirm that the elementary Tree-based, Bayesian and Rule-based classifiers along with Bagging benefit more from the use of Conditional Significance Vectors than the more complex classifiers like LogitBoost and Classification Via Regression and that IBk is not suitable at all for its use.

Thus, overall, the Reuters experiments conclude that the use of Conditional Significance Vectors improves the subtopic classification accuracy of the basic classifiers.

The LSHTC results again confirm that IBk is not suitable for semantic subspace learning and that LogitBoost and Classification Via Regression do not work very well for this purpose either. The LSHTC results also show the best subtopic classification accuracy with the maximum masking option for the remaining classifiers. As the LSHTC does not have overlapping subtopics, the vectors FS_9 and CS_9 are similar and the best LSHTC subtopic results are divided between these two vector formats.

In these experiments, the vector length of all the datasets derived from the same corpus is equal (50 for Reuters and 158 for LSHTC). We mask the different portions of a vector by setting the corresponding vector values to zero. However these zero values also contribute to the training of a classifier as they form a part of the input data pattern. Therefore, to remove certain subspaces from consideration, we need to remove all the document vector components corresponding to that subspace by deleting them instead of just masking them to zero value. However different main topics have different number of subtopics. For example, in our Reuters dataset, the CCAT main topic has 20 subtopics whereas the MCAT main topic has just 4 subtopics. The number of vector components required to represent different main topics (subspaces) is different whereas a single classifier learns on a fixed input vector length. Therefore our initial Semantic Subspace Learning architecture with a single classifier is not sufficient to deal with this case.

We need an architecture which has different classifiers to deal with different vector lengths of the separate subspaces. In case of today's vast data space, the subspaces can be as widely different from each other as medicine and politics and a single type of classifier may not be able to handle all the different types of subspaces equally well. Hence we modify this initial architecture to generate our final Hybrid Parallel Classifier architecture which applies different types of classifiers to different subspaces. In the Hybrid Parallel Classifier, we extract only the vector components relevant to a subspace thus reducing the length of the Conditional Significance Vector. We expect this to further improve classification accuracy as each classifier will deal with a reduced set of dimensions. In the subsequent experiments, we concentrate only on the basic classifiers as they have been shown to be suitable for semantic subspace learning. In the next chapter, we explain the experiments conducted and the results obtained by our final Hybrid Parallel Classifier architecture. We also measure the timing efficiency of the classifiers along with their classification accuracy in the next set of experiments.

Chapter 5

Hybrid Parallel Classifiers: Experiments & Results

In this chapter, we present the experimental evaluation of the Hybrid Parallel Classifier Architecture introduced in Chapter 3 (section 3.2.2). The Hybrid Parallel Classifier takes advantage of the different semantic subspaces existing in the data. At level 1, the Hybrid Parallel Classifier detects the relevant subspace of a document by using the Maximum Significance Value. This detection is done by comparing the numerical significance values of all the level 1 topics in the document vector. The detection time is O(k) where k is the number of level 1 topics. Subspace detection is therefore very fast and is independent of the total number of documents. At level 2, subtopic classification is done by the classifier best suited to the selected subspace. In this architecture, separate classifiers are trained on separate semantic subspaces using the Conditional Significance Vector with reduced dimensions. We conducted two sets of experiments with hybrid classifiers. In the first set of experiments (Experiment Set A), we determined the best classifier for each subspace and combined them together to form a hybrid classifier. We also tried to improve the performance of a multilayer perceptron (MLP) by combining it with other classifiers in various hybrid combinations. As the baseline for these experiments, we used the single MLP classifier on the full data space with two different vector formats, the Full Significance Vector and the tf-idf. The datasets used in this case were the Reuters Headlines and the LSHTC datasets. In the second set of experiments (Experiment Set B), we worked with a much wider variety of two-classifier and four-classifier hybrid combinations. We combined one type of basic classifier (e.g. Bayesian) with a basic classifier of another type (e.g. tree-based) in various two-classifier combinations and compared the classification accuracy of the basic classifier with all the two-classifier combinations in which this basic classifier also participated. We also experimented with various four-classifier combinations and compared their classification accuracies with those of their constituent classifiers. In these experiments, in addition to Reuters Headlines and LSHTC, we also used the Reuters Full Text dataset.

5.1 Experiment Set A: Hybrid Classifiers combining MLP with other types of basic classifiers

The experiments in Chapter 4 showed the Multilayer Perceptron (MLP) to be the best performing classifier in the majority of the cases. Hence, in these experiments, we attempted to improve upon the performance of the single MLP by combining it with other types of classifiers. These experiments used the Multilayer Perceptron (MLP) along with six other basic classification algorithms. These included two Bayesian algorithms (BayesNet & Naive Bayes), two rule-based algorithms (PART & NNge) and two tree-based algorithms (J48 &

Random Forest). Our experiments were run using these seven algorithms from Weka on the Reuters Headlines and LSHTC datasets. The experimental setup was as follows:

- Convert the text datasets into the Conditional Significance Vector representation.
- Train / Test Split taken as 9000 training vectors / 1000 test vectors for Reuters Headlines dataset and 4000 training vectors / 463 test vectors for the LSHTC dataset.
- Training vectors further divided into different subsets according to their main topics.
- Training data subsets used to separately train the classifiers for each main topic (subspace).
- For each subspace, only the vector dimensions representing the subtopics of that main topic were extracted from the complete document vector.
- For a test vector, the main topic was identified by the Maximum Significance Value. Here, all the main topic vector entries were inspected and the maximum among these values was determined. The main topic corresponding to this maximum value was taken as the main topic of the test document.
- The subtopic vector entries corresponding to this predicted main topic were extracted along with their *actual* subtopic label and given to the classifier trained for this main topic (subspace).

Determination of classifier to be allocated for each subspace:

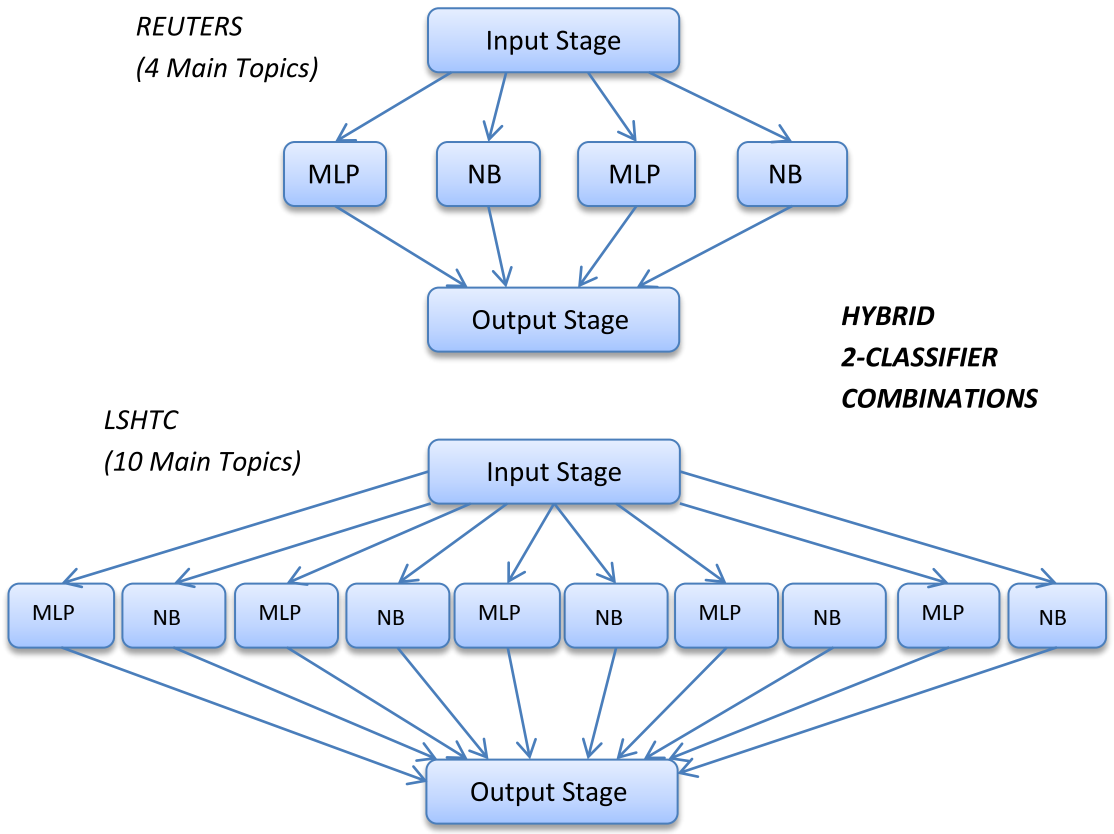
This was done in two different ways as follows:

i. Experimental Determination:

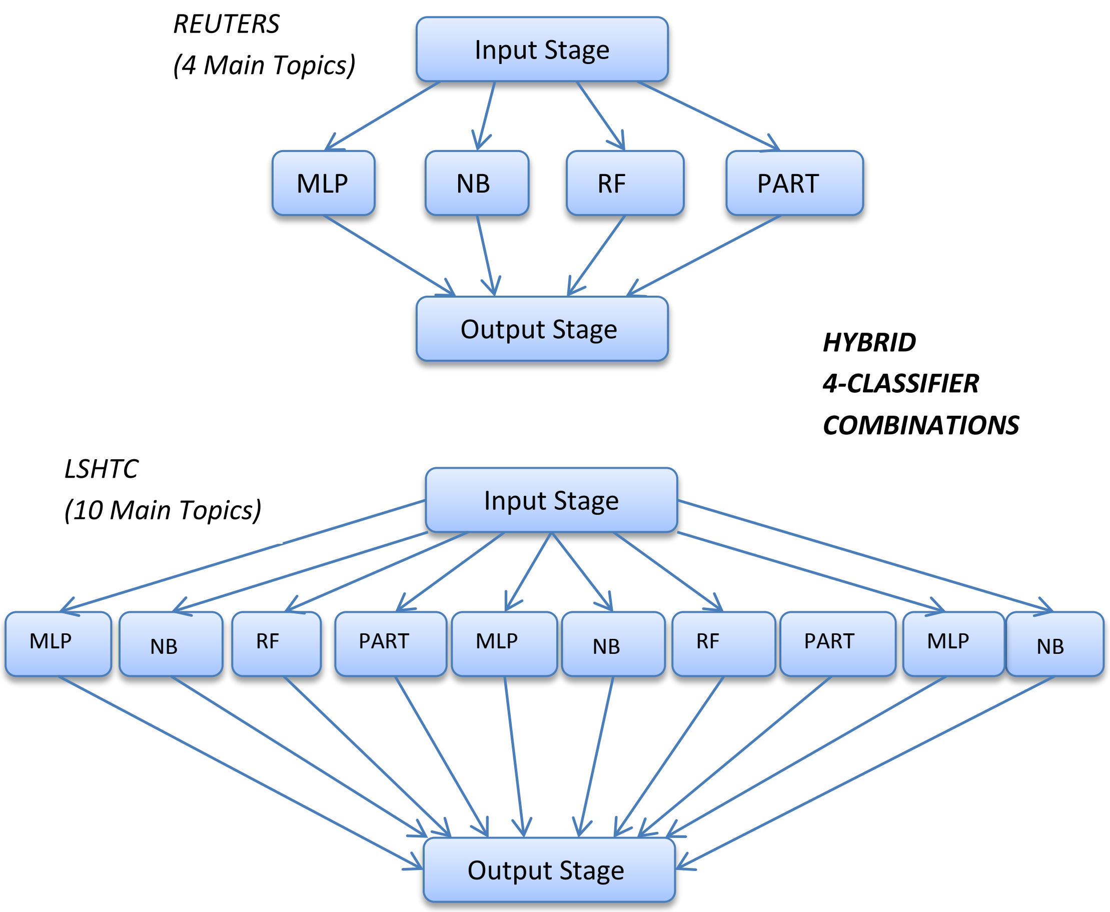
In this case, we used the category-wise separated data from the training set to select the algorithm with the highest classification accuracy for each main category. In the case of a tie between two algorithms, the one with the lower training time was chosen.

ii. Predefined Combinations

As the performance of many classifiers for each main category was quite close to each other, we also ran some experiments with hybrid classifiers using a predefined combination of basic classifiers. Here, the MLP was combined with different types of classifiers (Bayesian, rule-based and tree-based classifiers) in various two-classifier and four-classifier combinations. For a two-classifier combination, MLP and the other classifier were used alternately on the main category topics of the Reuters and LSHTC datasets as follows:



For a four-classifier system four different classifiers were used on the four main topics of Reuters Headlines dataset and repeated for each block of four main topics for the LSHTC dataset as follows:

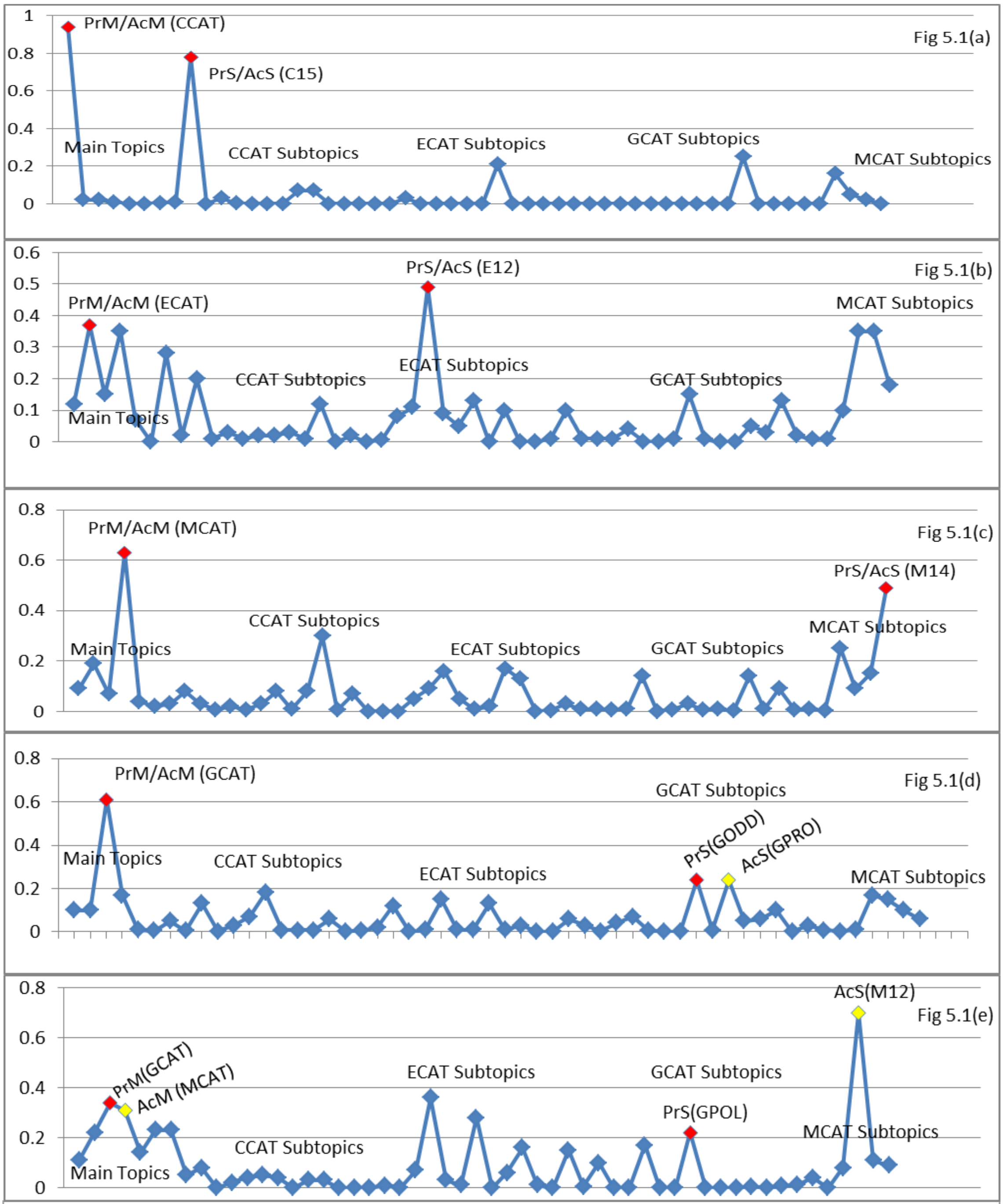


Subsequently we applied these selected algorithms to the test data and measured the performance of the hybrid classifier. The category-wise separated Conditional Significance Vectors were used here. We also ran the MLP classifier on the full (not category-wise separated) data set to provide a comparison for the hybrid classifier. Two vector representations were used for the comparison baseline – the Full Significance Vector and tf-idf. The combination of MLP with different types of classifiers (Bayesian, rule-based and tree-based classifiers) was evaluated and the best combination was identified.

5.1.1 Upper Limit on Hybrid Classifier Accuracy

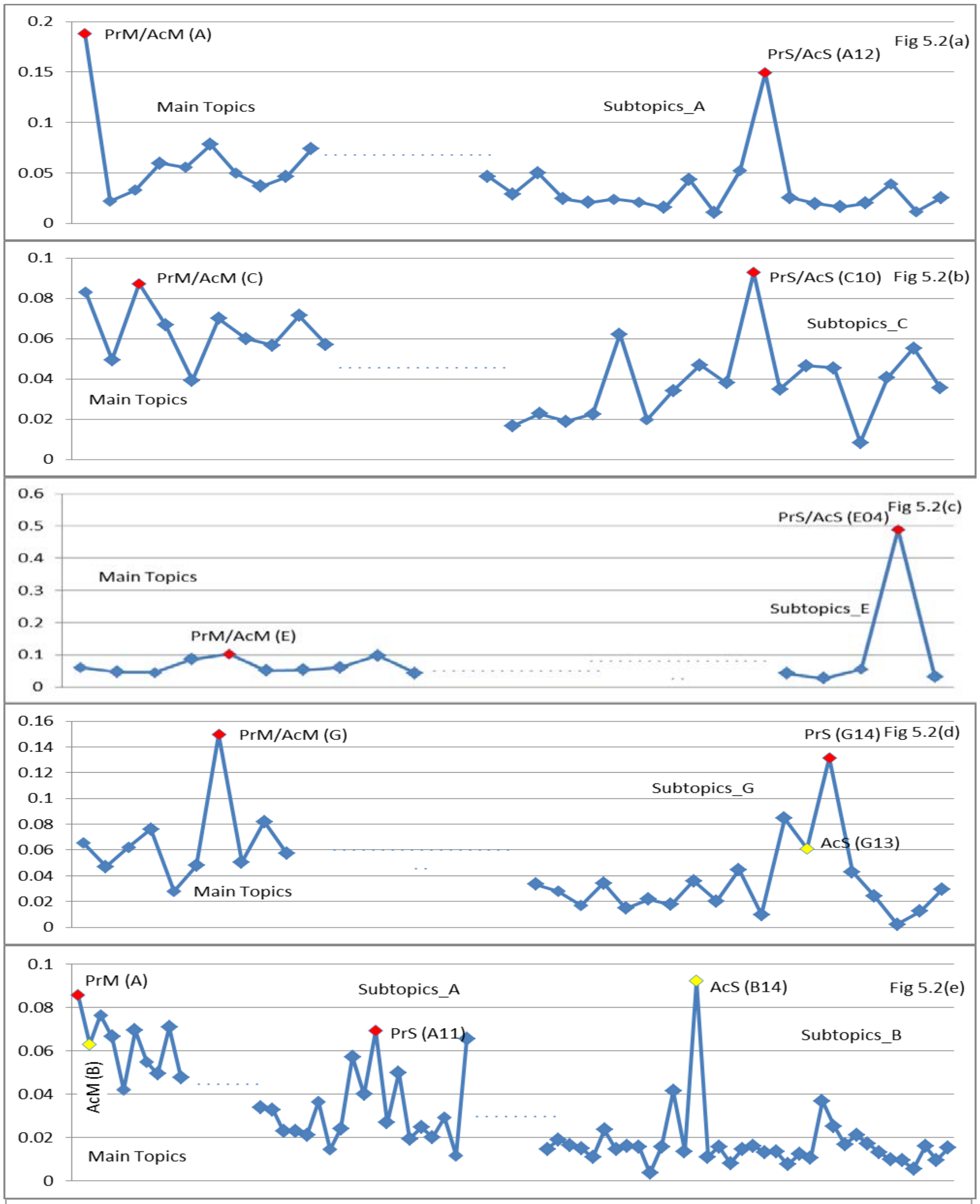
Figure 5.1 shows the classification decisions for a set of Reuters Headlines input vectors by a hybrid classifier. Figures 5.1(a) – 5.1(e) each represent one input test vector. The x-axis of these figures represents the significance vector components which in turn represent all the main topics and subtopics present in our Reuters Headlines data. The y-axis shows the actual numerical values for these significance vector components as calculated in sections 3.1.2 and 3.1.3. The red data points show the predicted main topic and the predicted subtopic while the yellow data points show the actual main topic and the actual subtopic (wherever actual and predicted are distinct). Figures 5.1(a), 5.1(b) and 5.1(c) show correctly classified vectors while Figures 5.1(d) and 5.1(e) show vectors which are classified wrongly. In Figures 5.1(a), 5.1(b) and 5.1(c), there are no yellow data points as the predicted and actual main topics are the same. In Figure 5.1(d), the main topic predicted was correct and the vector was presented to the correct classifier but the subtopic classification was wrong. Hence the figure shows red and yellow data points for the subtopic. In Figure 5.1(e), the main topic predicted was wrong and hence the vector was presented to the wrong classifier – resulting in a wrong classification. This figure shows red and yellow data points for both the main topic as well as the subtopic. Figure 5.1(e) presents an inherent limitation of this system whereby a wrong classifier is chosen by the classifier selection step of the parallel classifier.

For the Reuters Headlines, the accuracy of choosing the correct main topic by selecting the maximum significance level 1 entry was measured to be 96.80% for the 1000 test vectors, i.e. 968 vectors were assigned the correct trained classifiers whereas 3.20% or 32 vectors were assigned to a wrong classifier – resulting in a wrong classification decision for all these 32 vectors. Hence the upper limit for classification accuracy was 96.80% for our hybrid parallel classifier for the Reuters Headlines dataset.



PrM – Predicted Main Topic PrS – Predicted Subtopic
AcM – Actual Main Topic AcS – Actual Subtopic
Red data points show the predicted main topic and the predicted subtopic
Yellow data points show the actual main topic and the actual subtopic

Fig 5.1: Classification Decisions by a Hybrid Parallel Classifier for some REUTERS Headlines Input Vectors



PrM – Predicted Main Topic PrS – Predicted Subtopic
AcM – Actual Main Topic AcS – Actual Subtopic
Red data points show the predicted main topic and the predicted subtopic
Yellow data points show the actual main topic and the actual subtopic

Fig 5.2: Classification Decisions by a Hybrid Parallel Classifier for some LSHTC Input Vectors

Similarly, the accuracy of choosing the correct main topic by selecting the maximum significance level 1 entry was measured to be 85.31% for the 463 LSHTC test vectors, i.e. 85.31% or 395 vectors were assigned the correct trained classifiers whereas 14.69% or 68 vectors were assigned to a wrong classifier – resulting in a wrong classification decision for all these 68 vectors. Hence the upper limit for classification accuracy was 85.31% for our hybrid parallel classifier for the LSHTC dataset. Figures 5.2(a), 5.2(b) and 5.2(c) show relevant snapshots of the correctly classified LSHTC vectors while Figures 5.2(d) and 5.2(e) show snapshots of the LSHTC vectors which are classified wrongly.

5.1.2 Hybrid Parallel Classifier Results for Experiment Set A

The graphs in Figure 5.3 show a comparison of the performance of hybrid classifiers with that of MLP for both corpora. The subtopic classification accuracy percentage and training time in seconds is shown for the Hybrid Parallel classifiers along with that of the baselines. The baseline was a single MLP classifier using full data (not category-wise separated data). This baseline experiment was run with two different vector representations – Significance Vector and tf-idf. The accuracies of all the hybrid parallel classifiers were better than that of the single MLP classifier. This was due to the fact that each base classifier present in the hybrid parallel classifier had to learn from a subset of the original data. As such, it was able to distinguish between categories present in this subspace more accurately than a classifier which had to learn from the full dataset.

Overall, it was observed that there was an improvement in subtopic classification accuracy along with a significant reduction in training time. The classification accuracies of all the hybrid classifiers were quite close to each other but all of them were much better than the classification accuracy of the single classifier with the tf-idf baseline for both the Reuters Headlines and the

LSHTC datasets. The difference with the significance vector baseline was less for Reuters Headlines but even there the classification accuracies of the hybrid systems were better. The training times showed a very steep reduction compared to both baselines. The average of 10 runs was taken for each experiment. In the hybrid classifier, even though we are using more classifiers, the training time is reduced. This is because each classifier was now trained on a reduced set of data with a reduced set of vector components. This two-fold reduction translates to a significant decrease in training time.

We also compared the performance of one hybrid classifier (HC4) with three different vector formats: FSV_FullVector, FSV_RelVector and CSV_RelVector. The FSV_FullVector was the complete Full Significance Vector while FSV_RelVector was the Full Significance Vector with only the relevant subtopic columns corresponding to the a document’s main topic. Similarly, CSV_RelVector was the Conditional Significance Vector with only the relevant subtopic columns corresponding to a main topic. It was observed that the CSV_RelVector gave the highest subtopic classification accuracy. The use of only relevant dimensions reduced the document vector length to be handled by the constituent basic classifiers. This improved the effectiveness of these basic classifiers. The use of Conditional Significance further enhanced the distinction between subtopics within a subspace. The combination of these two factors led to the best classification result being obtained by the CSV_RelVector

Reuters Corpus Results for Experiment Set A:

Figure 5.3 shows the results for hybrid classifiers for the Reuters Headlines dataset with the two baselines (MLP with Full Significance Vector and and MLP with tf-idf vector). Figures 5.4(a) and 5.4(b) show the detailed Reuters Headlines results for the hybrid classifiers. The Hybrid 4-classifier system (HC10) showed the best classification accuracy which was quite similar to that of the hybrid classifier with category-wise classifiers chosen from the training set (HC1). The training times of all the hybrid classifiers were quite close to each other with HC1, HC8, HC9 and HC10 showing the least training time. The other hybrid classifiers were two-classifier systems with one MLP and one non-MLP classifier alternating on the main topics. Therefore, the Reuters main topics CCAT/ECAT/GCAT/MCAT had MLP/Non-MLP/MLP/Non-MLP as the corresponding basic subspace classifiers. Hence for the Reuters Headlines data there were two MLPs in all the hybrid 2-classifier systems. This could account for the slightly higher training time of these classifiers versus the hybrid 4-classifier systems (HC8, HC9 and HC10) which had only one MLP in the combination. The hybrid classifier with category-wise classifiers chosen from training set (HC1) had MLP for the CCAT main topic and J48 for all other main topics. Since this combination also had only one MLP, its training time was comparable to the hybrid 4-classifier systems.

Figure 5.5(a) shows the comparison of the classification accuracy of the best hybrid classifier (HC10) on category-wise data with that of each basic classifier on full data for the Reuters Headlines dataset. The average basic classifier accuracy is also shown. The chart shows the performance of each basic classifier using two different vector formats – tf-idf and Significance Vector. The performance of the hybrid classifier was better than the average basic classifier accuracy for both vector formats. The hybrid classifier comprised of four different classifiers operating on the four subspaces of the Reuters Headlines dataset. Each classifier within the hybrid classifier was more effective as it had to distinguish only between subtopics within a main topic. The performance of each classifier within the combination was thus improved leading to an overall higher performance of the hybrid classifier. Each basic single classifier, on the other hand, had to distinguish between all subtopics of all main topics in the Reuters Headlines dataset. As this was a very large number, the performances of the various basic single classifiers were reduced leading to a lower average basic classifier accuracy.

Figures 5.6(a) and 5.6(b) show the performance of the HC4 classifier (Hybrid parallel 2-classifier MLP/NNge combination) with different vector formats for the Reuters Headlines dataset. It can be seen that CSV_RelVector (Conditional Significance Vectors with only the relevant subtopic vector components) gave the highest subtopic classification accuracy and the lowest training time. Relevant subtopic vector components reduced the vector length handled by a subspace classifier and the use of Conditional Significance improved the distinction between subtopics within a subspace. The combination of these two factors lead to the highest performance of the CSV_RelVector.

LSHTC Corpus Results for Experiment Set A:

Figure 5.3 also shows the results of the Hybrid Classifiers for the LSHTC dataset with both baselines (MLP with Full Significance Vector and MLP with tf-idf vector). Figures 5.4(c) and 5.4(d) show the detailed results for the LSHTC hybrid classifiers. The highest subtopic classification accuracy was shown by the Hybrid Parallel Classifier with category-wise classifiers chosen from training data performance (HC1) with 82.85%. It had a training time of 63.69 seconds. This was very closely followed by Hybrid 2-Classifier (MLP/NNge) System (HC4) with 82.72% classification accuracy and 43.68 seconds training time. The lowest training time was shown by the Predefined Hybrid 4-Classifier System (MLP/NB/NNge/J48) (HC8) at 24.14 seconds. In an overall tradeoff between classification accuracy and training time (almost best accuracy and much less training time), the best hybrid classifier seemed to be the Hybrid 2-Classifier System (MLP/NNge) (HC4). This classifier also eliminated the step of choosing the best classifier per main category from the training set and thus effectively reduced training time even further.

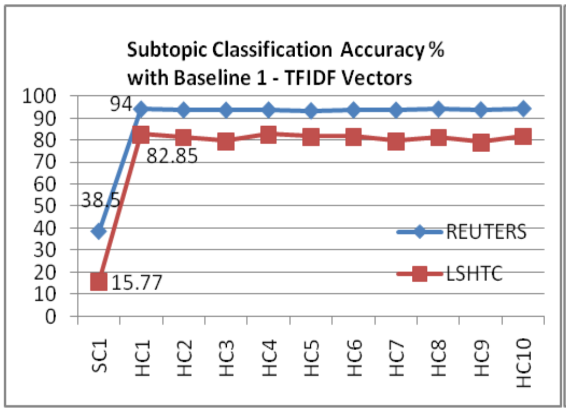


Fig 5.3(a)

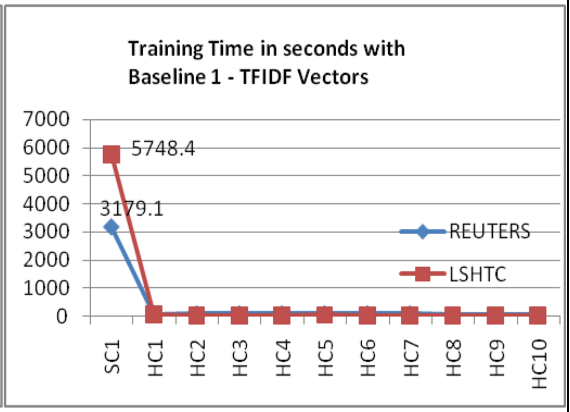


Fig 5.3(b)

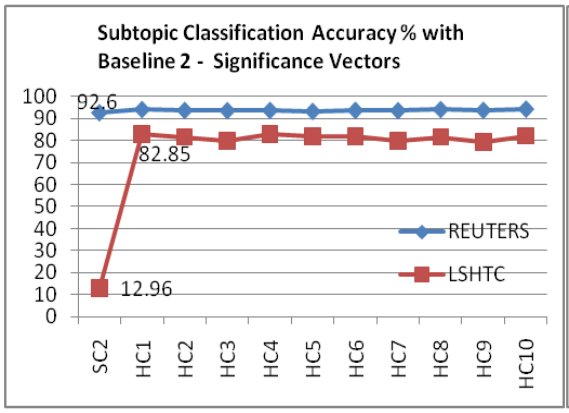


Fig 5.3(c)

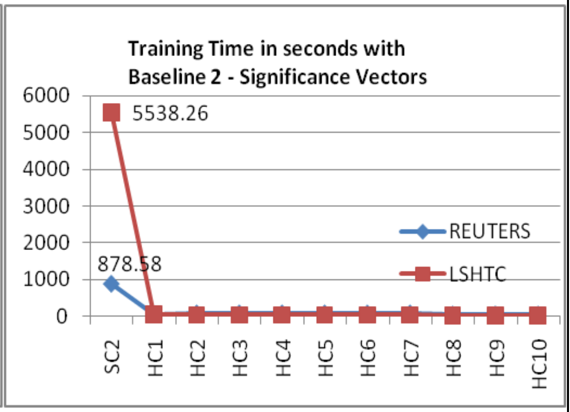


Fig 5.3(d)

Classifier Index

SC1- Single MLP over full data using tf-idf Vectors
SC2- Single MLP over full data using Significance Vectors
HC1- Hybrid Parallel Classifier with category-wise classifiers chosen from training data performance
HC2- Hybrid 2-Classifier System (MLP/NB)* HC5- Hybrid 2-Classifier System (MLP/PART)*
HC3- Hybrid 2-Classifier System (MLP/BN)* HC6- Hybrid 2-Classifier System (MLP/J48)*
HC4- Hybrid 2-Classifier System (MLP/NNge)* HC7- Hybrid 2-Classifier System (MLP/RF)*

HC8- Hybrid 4-Classifier System (MLP/NB/ NNge/ J48)*
HC9- Hybrid 4-Classifier System (MLP/BN/PART/RF)*
HC10- Hybrid 4-Classifier System (MLP/NNge/PART/NB)*

*MLP - Multilayer Perceptron (Neural Network); NB - Naïve Bayes, BN - BayesNet (Bayesian);
NNge - Nearest Neighbour with Generalised Exemplars, PART - PART Decision List (Rule Based);
J48 - Weka's version of C4.5, RF - Random Forest (Tree Based);

Fig 5.3: Hybrid Parallel Classifiers Performance Metrics with MLP Baselines

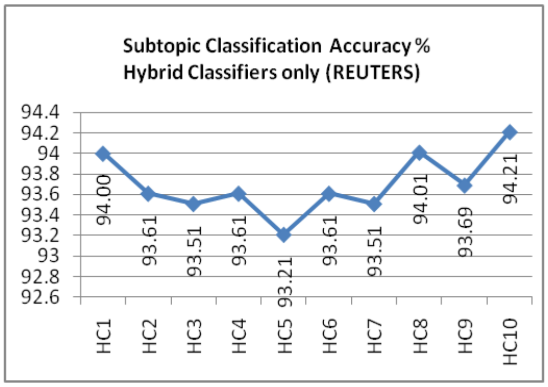


Fig 5.4(a)

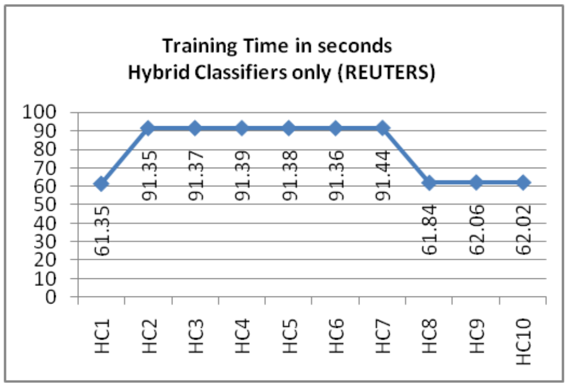


Fig 5.4(b)

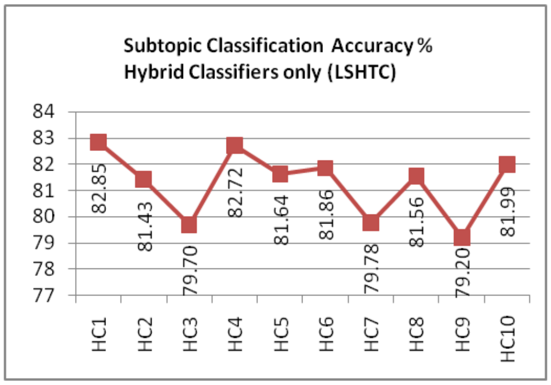


Fig 5.4(c)

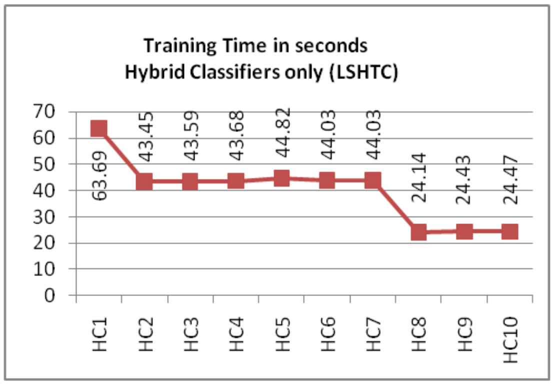


Fig 5.4(d)

Classifier Index

HC1- Hybrid Parallel Classifier with category-wise classifiers chosen from training data performance
HC2- Hybrid 2-Classifier System (MLP/NB)*
HC3- Hybrid 2-Classifier System (MLP/BN)*
HC4- Hybrid 2-Classifier System (MLP/NNge)*
HC5- Hybrid 2-Classifier System (MLP/PART)*
HC6- Hybrid 2-Classifier System (MLP/J48)*
HC7- Hybrid 2-Classifier System (MLP/RF)*
HC8- Hybrid 4-Classifier System (MLP/NB/ NNge/ J48)*
HC9- Hybrid 4-Classifier System (MLP/BN/PART/RF)*
HC10- Hybrid 4-Classifier System (MLP/NNge/PART/NB)*

*MLP - Multilayer Perceptron (Neural Network); NB - Naïve Bayes, BN - BayesNet (Bayesian);
NNge - Nearest Neighbour with Generalised Exemplars, PART - PART Decision List (Rule Based);
J48 - Weka's version of C4.5, RF - Random Forest (Tree Based);

Fig 5.4: Performance Metrics - Hybrid Parallel Classifiers Only

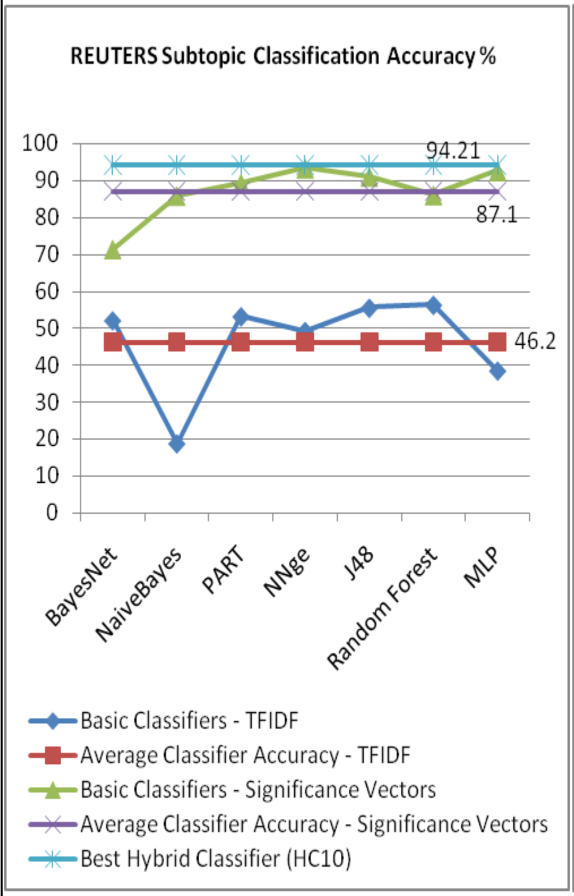


Fig 5.5(a)

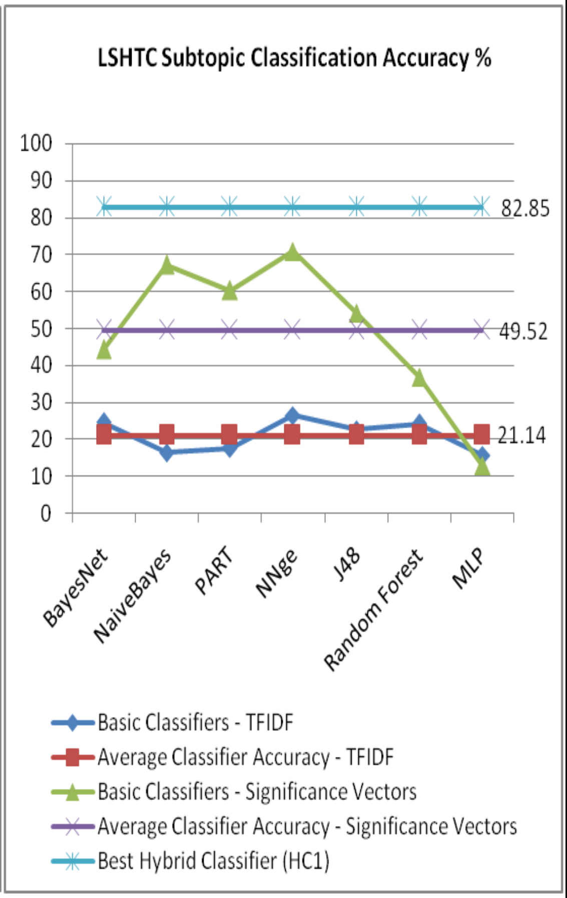


Fig 5.5(b)

Fig 5.5: Comparison of Hybrid Classifier Performance with Basic Classifiers on Full Data space

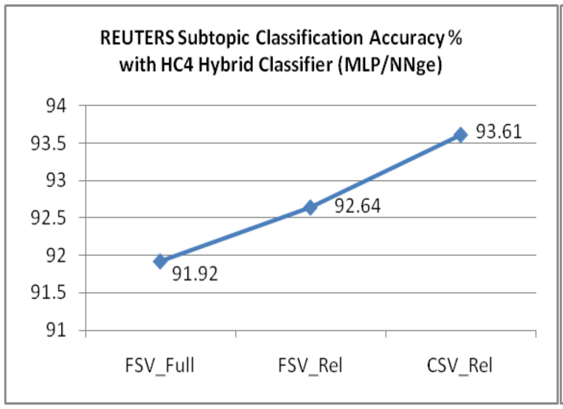


Fig 5.6(a)

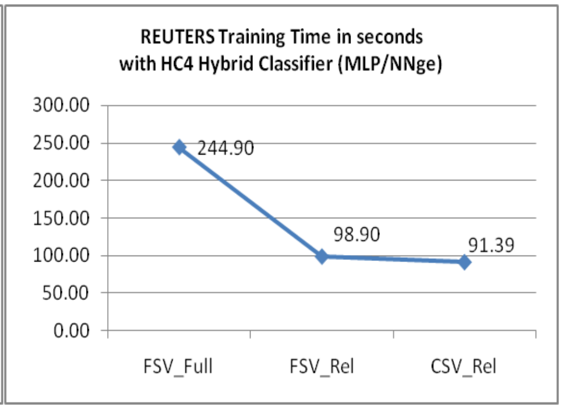


Fig 5.6(b)

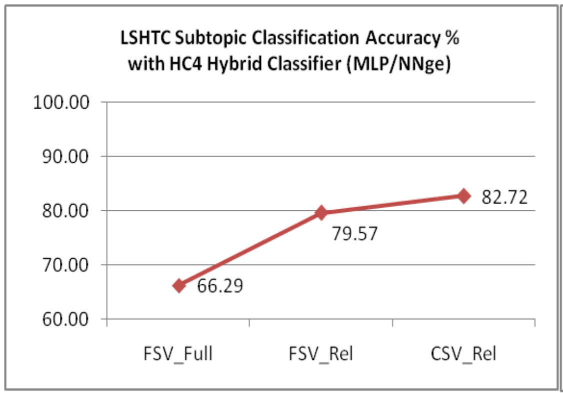


Fig 5.6(c)

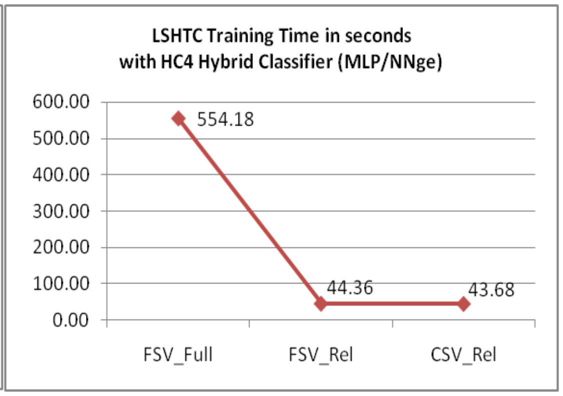


Fig 5.6(d)

HC4: Hybrid Parallel 2-Classifier Combination with MLP and NNge

Vector Formats:

FSV_Full: Full Significance Vector with the full set of subtopic vector components
(Vector Length is 50 for Reuters and 158 for LSHTC)

FSV_Rel : Full Significance Vector with only the set of subtopic vector components relevant to the Main Topic

CSV_Rel : Conditional Significance Vector with only the set of subtopic vector components relevant to the Main Topic

Fig 5.6: Comparison of Hybrid Classifier (HC4) Performance with different Vector Formats

Figure 5.5(b) shows the comparison of the classification accuracy of the best hybrid classifier (HC1) on category-wise data with that of each basic classifier on full data for the LSHTC dataset. The average classification accuracy is also shown. The chart shows the performance of each basic classifier using two different vector formats – tf-idf and Significance Vector. The performance of the hybrid classifier was much better than the average basic classifier accuracy for both vector formats. Though the numerical values of classification accuracies are higher with the Reuters Headlines dataset, the *improvement* in performance is much higher with the LSHTC dataset.

Figures 5.6(c) and 5.6(d) show the performance of the HC4 classifier (Hybrid parallel 2-classifier MLP/NNge combination with different vector formats for the LSHTC dataset. Here again, it can be seen that CSV_RelVector (Conditional Significance Vectors with only the relevant subtopic vector components) gives the best subtopic classification accuracy and training time. Here again, the improvement obtained was higher with the LSHTC dataset than with the Reuters Headlines dataset. The main reason for this is that the baseline FSV_FullVector had 158 components for LSHTC and only 50 components for Reuters. As the LSHTC had 10 main topics, the average number of relevant vectors components in FSV_RelVector and CSV_RelVector was 158/10 or 15.8 for LSHTC. Similarly the Reuters datasets had 4 main topics and thus an average number of 50/4 or 12.5 relevant vector components for each subspace. Thus the reduction in the number of vector components was much greater with LSHTC than Reuters leading to the higher improvement in classification accuracy and training time with the LSHTC dataset. The use of Conditional Significance improved the distinction between subtopics within a main topic thus causing further improvement with the use of CSV_RelVector.

Numerically the hybrid classifier accuracy values obtained for the Reuters Headlines dataset were greater than the hybrid classifier accuracy values obtained for the LSHTC dataset (93.21% – 94.21% for Reuters Headlines v/s

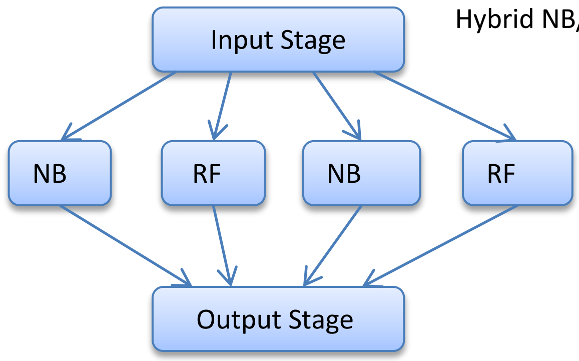
79.20% - 82.85% for LSHTC Corpus). However, the *improvement* in performance over the baselines was much more marked with the LSHTC dataset compared to the Reuters Headlines dataset. Again, with the different vector formats, improved performance was found with the Reuters dataset, but the improvement obtained was higher with LSHTC. As the LSHTC dataset has more categories (10 main and 158 subtopic) than the Reuters Headlines dataset (4 main and 50 subtopics), this result is particularly encouraging.

5.2 Experiment Set B: Hybrid Classifiers combining a variety of basic classifiers

These experiments used six classification algorithms, namely Random Forest, J48 (C4.5), the Multilayer Perceptron, Naïve Bayes, BayesNet and PART. The test corpora used were the LSHTC dataset, the Reuters Headlines dataset and the Reuters Full Text dataset. The Reuters Full Text dataset was formed by merging the headlines and body text of each news item. Figure 5.7 shows an example of Reuters Full Text.

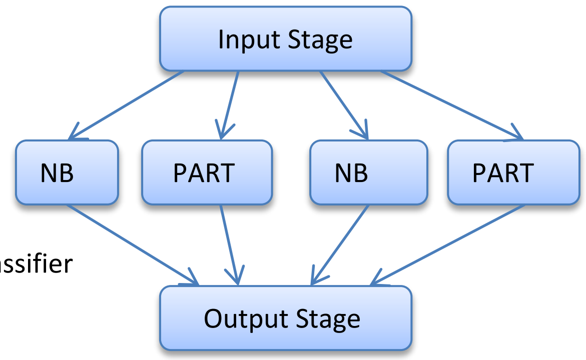
We tested various hybrid 2-classifier and 4-classifier combinations. The experimental setup was as follows:

- For the hybrid 2-classifier combinations, a classifier of one type was combined with classifiers of other types in a large variety of combinations. Some examples of NB-based hybrid combinations for the Reuters dataset with 4 main topics are shown below:



Hybrid NB/RF Classifier

HYBRID 2-CLASSIFIER COMBINATIONS WITH NAÏVE BAYES (NB)



Hybrid NB/PART Classifier

- The performance of each single classifier on the full data was compared with the performance of the hybrid 2-classifier combinations in which this particular classifier also participated. In the NB-based hybrid combinations shown above the comparison baseline would be the performance of the single NB classifier on the full data space.
- For the single classifier experiments, the Full Significance Vector representation was used, whereas for the hybrid classifier experiments, the category-wise separated Conditional Significance Vector representation was used.
- For Hybrid 4-Classifier combinations, four different types of classifiers were used for the four main topics of the Reuters Corpus. A variety of 4-classifier combinations were tested here. For the LSHTC Corpus, the 4-Classifier combination was repeated after every block of four main topics.

- The data was divided into 9000 training/1000 test vectors for the Reuters Headlines as well as the Reuters Full Text datasets and 4000 training/463 test vectors for the LSHTC dataset.
- The result values represent an average of 10 runs with different parameter values.

<u>Example 1</u>

<u>Headline:</u>

Planet Hollywood launches credit card

<u>Body Text:</u>

If dining at Planet Hollywood made you feel like a movie star now you can spend money like Arnold Schwarzenegger with a new credit card from the themed restaurant chain.

The fast growing company whose outlets are festooned with kitsch movie memorabilia has teamed up with the William Morris talent agency and MBNA America Bank of Wilmington-Del to offer a credit card with appropriate Hollywood perks.

These include preferential seating in the restaurants, a limited edition T-shirt and discounts on food and merchandise, a statement said. Planet Hollywood joins other pop culture companies such as Rolling Stone magazine that are issuing branded credit cards that make going into debt more fun than usual.

Approved applicants don't have to pay an annual fee and there's a special introductory annual percentage rate of 5.9 percent for balance transfers and cash advance checks. Orlando, Florida based Planet Hollywood is part of Planet Hollywood International Inc.

<u>Full Text = Headline + Body Text</u>

Fig 5.7: An Example of Reuters Full Text

5.2.1 Reuters Results for Experiment Set B

In all combinations, it was observed that hybrid 2-classifier combinations performed better than the single basic classifier. Figure 5.8 shows the subtopic classification accuracy of the hybrid 2-classifier combinations along with the subtopic classification accuracy of single basic classifiers for both the Reuters Headlines as well as the Reuters Full Text datasets. Both the datasets followed a similar pattern where all the hybrid classifiers performed better than any of the single classifiers. In both cases, this was statistically significant (Wilcoxon Signed Rank h = 1, p = 1.304e-05). Numerically, the classification accuracy values for the Reuters Headlines were higher than those of Reuters Full Text.

The single classifier performances also showed a similar pattern for both datasets. In the tree based classifiers, J48 performed better than Random Forest for Reuters Headlines and vice-versa for Reuters Full Text. In Figure 5.8, the hybrid classifier data points immediately above a particular single classifier show the 2-classifier combinations which include that single classifier e.g. the hybrid classifier data points H1-H4, which are above the single classifier Naïve Bayes, show the two-classifier combinations which include Naïve Bayes. As can be seen in the figure all the hybrid 2-classifier combinations performed better than the corresponding single classifiers.

Figure 5.9 shows the subtopic classification accuracy of the hybrid 4-classifier combinations along with the subtopic classification accuracy of single basic classifiers for both the Reuters Headlines as well as the Reuters Full Text datasets. Here again, both datasets followed a similar pattern where all the hybrid classifiers performed better than any of the single classifiers. Once again, this was statistically significant for both headlines and full text (Wilcoxon Signed Rank h = 1, p = 0.03125). In this case too, the numerical classification accuracy values for the Reuters headlines were higher than those of Reuters Full Text. Possible reasons for this will be discussed in the conclusion to this chapter.

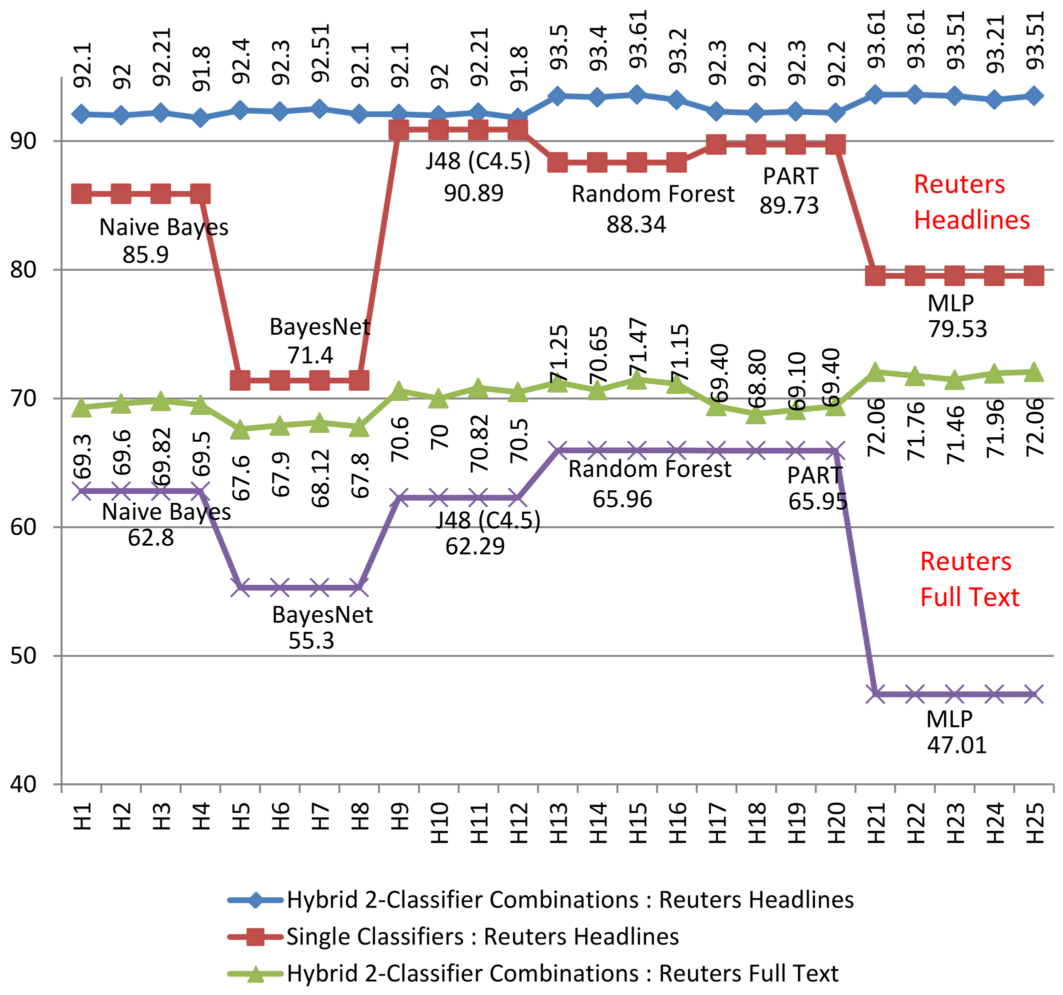
Upper Limit for Reuters Full Text Hybrid Classifier

Similar to the Reuters Headlines and LSHTC datasets, we checked the accuracy of choosing the correct main topic by selecting the maximum significance level 1 entry for the Reuters Full Text dataset. It was found to be 82.50% for the 1000 test vectors. Hence, this is the upper limit for the hybrid parallel classifier for the Reuters Full Text dataset.

5.2.2 LSHTC Results for Experiment Set B

The results for the LSHTC dataset also showed that all the two-classifier and four-classifier combinations performed better than the single classifiers. Figure 5.10 and Figure 5.11 show the LSHTC results. The single classifier performances also showed a similar pattern to the Reuters datasets. These results were again statistically significant for two-classifier combinations (Wilcoxon Signed Rank h = 1, p = 1.2290e-005) as well as four-classifier combinations (Wilcoxon Signed Rank h = 1, p = 0.03125).

In the tree based classifiers, J48 performed better than Random Forest as in Reuters Headlines. Similar to the Experiment Set A (section 5.1), the improvement in performance by using hybrid classifiers was much more marked with the LSHTC dataset as compared to both the Reuters Headlines as well as the Reuters Full Text dataset. For example, the hybrid two-classifier combinations containing J48 showed an improvement of about 1.5% for Reuters Headlines, 8% for Reuters Full Text and 25% for LSHTC over the corresponding baseline single J48 classifiers. Similarly, the hybrid two-classifier combinations containing Naïve Bayes showed an improvement of about 6% for Reuters Headlines, 6% for Reuters Full Text and 13% for LSHTC over the corresponding baseline single Naïve Bayes classifiers. The BayesNet classifier similarly showed an improvement about 21% for Reuters Headlines, 12% for Reuters Full Text and 35% for the LSHTC dataset on hybridization. As the LSHTC dataset has more categories than the Reuters datasets, this shows that the effectiveness of Hybrid Classifiers increases with the increasing number of categories. We will discuss possible reasons for this in the conclusion of this chapter.



<ins>Hybrid 2-Classifier Combinations</ins>:

H1 - NB/J48	H9 - J48/NB	H17 - PART/NB	H21 - MLP/NB
H2 - NB/RF	H10 - J48/BN	H18 - PART/BN	H22 - MLP/J48
H3 - NB/MLP	H11 - J48/MLP	H19 - PART/J48	H23 - MLP/BN
H4 - NB/PART	H12 - J48/PART	H20 - PART/RF	H24 - MLP/PART
H5 - BN/J48	H13 - RF/NB		H25 - MLP/RF
H6 - BN/RF	H14 - RF/BN	NB - Naive Bayes	
H7 - BN/MLP	H15 - RF/MLP	BN - BayesNet	
H8 - BN/PART	H16 - RF/PART	RF - Random Forest	

Fig 5.8: Subtopic Classification Accuracy for Hybrid Two-Classifier Combinations for the REUTERS Corpus

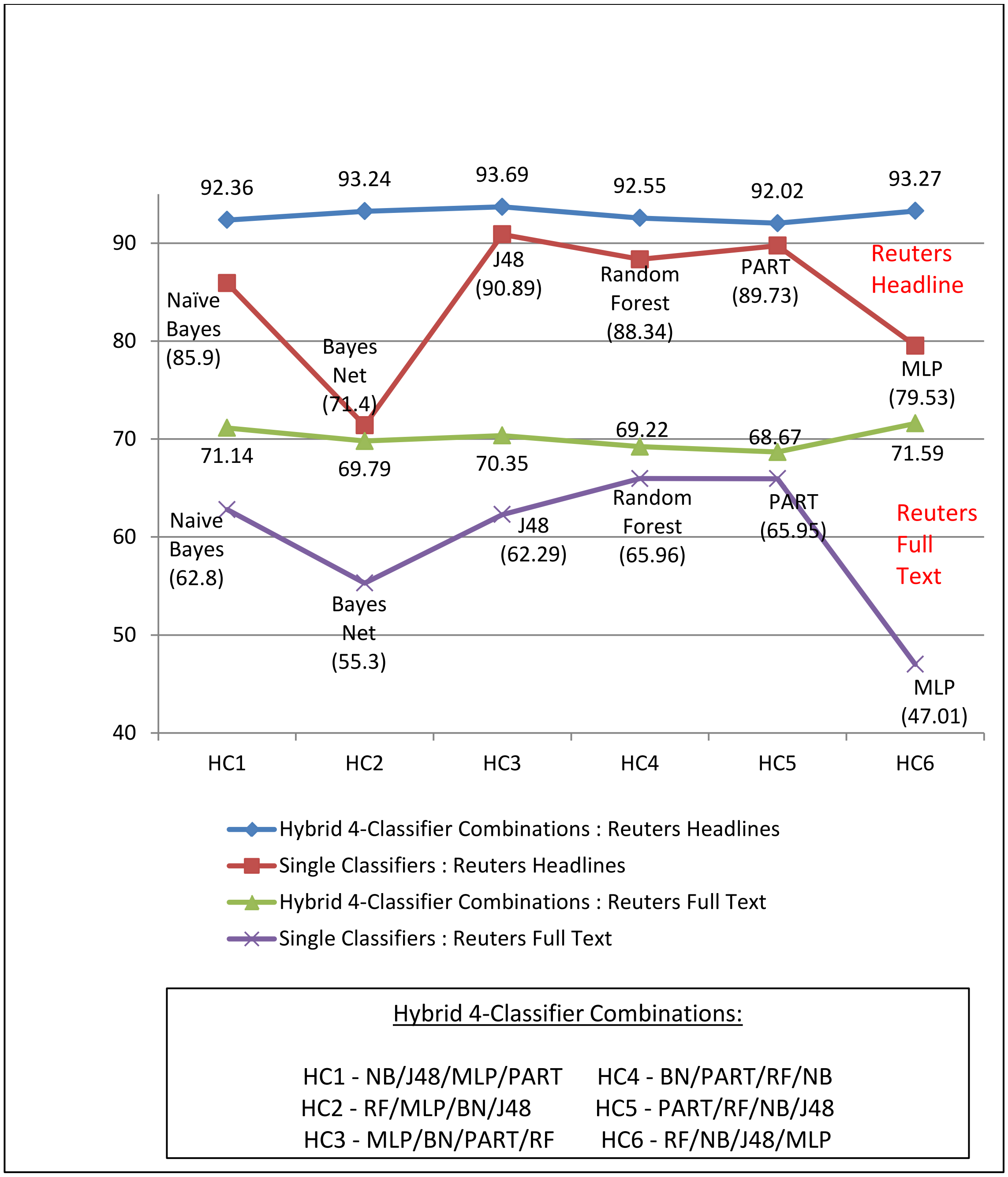
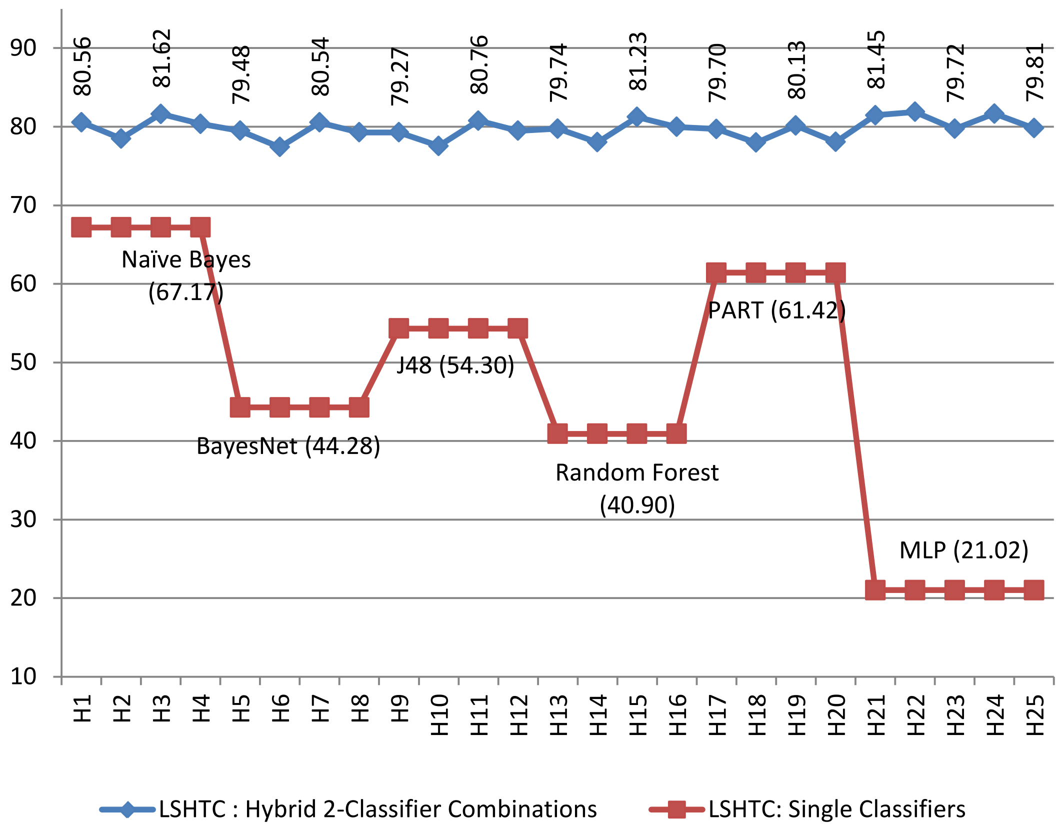


Fig 5.9: Subtopic Classification Accuracy for Hybrid Four-Classifier Combinations for the REUTERS Corpus



<u>Hybrid 2-Classifier Combinations:</u>

H1 - NB/J48	H9 - J48/NB	H17 - PART/NB	H21 - MLP/NB
H2 - NB/RF	H10 - J48/BN	H18 - PART/BN	H22 - MLP/J48
H3 - NB/MLP	H11 - J48/MLP	H19 - PART/J48	H23 - MLP/BN
H4 - NB/PART	H12 - J48/PART	H20 - PART/RF	H24 - MLP/PART
			H25 - MLP/RF
H5 - BN/J48	H13 - RF/NB		
H6 - BN/RF	H14 - RF/BN		
H7 - BN/MLP	H15 - RF/MLP		
H8 - BN/PART	H16 - RF/PART		

Fig 5.10: Subtopic Classification Accuracy for Hybrid Two-Classifier Combinations for the LSHTC Corpus

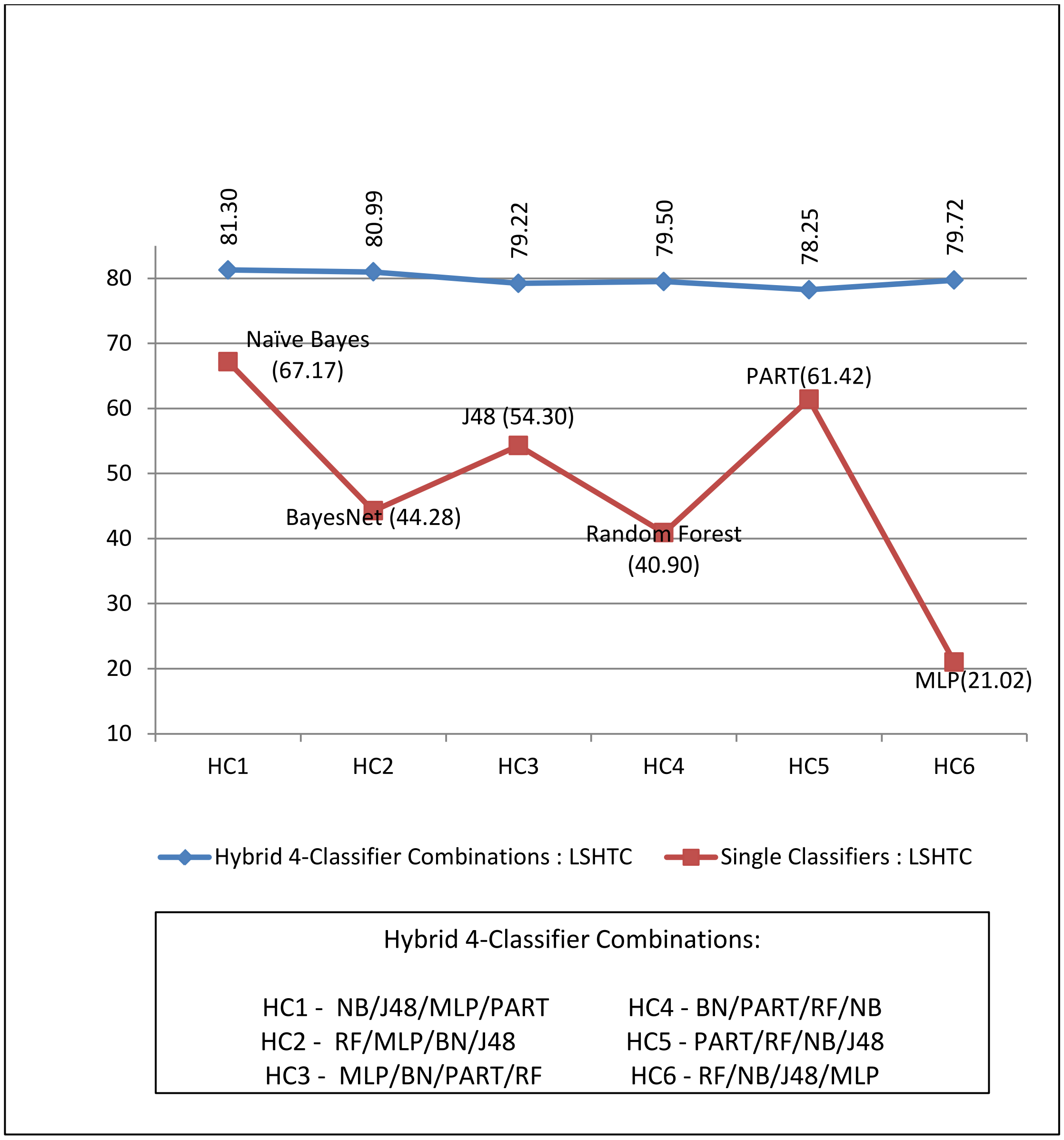


Fig 5.11: Subtopic Classification Accuracy for Hybrid Four-Classifier Combinations for the LSHTC Corpus

5.3 Conclusion

In these experiments, we attempted to leverage the differences in the characteristics of different subspaces to improve semantic subspace learning.

The main objective was to improve document classification in a vast document space by combining various learning methods. Our results with experiment set A (section 5.1) showed that combining MLP in parallel with a basic classifier (Bayesian, tree based or rule based) improved the classification accuracy and significantly reduced the training time compared to using a single MLP on the full data space. The performance improvement was even more significant when the number of main topics and subtopics was large (LSHTC v/s Reuters). The results with experiment set B (section 5.2) showed that combining a basic classifier in parallel with classifiers of other types in a hybrid combination increased the classification accuracy of the basic classifier concerned. They also showed that combining various types of classifiers in a hybrid combination resulted in a classification accuracy better than that of all the constituent single classifiers. The Wilcoxon Signed Rank test conducted on these results showed them to be statistically significant. Similar to Experiment Set A, these results also showed a higher improvement with the LSHTC dataset than with both the Reuters datasets. As the LSHTC dataset had 10 main topics, the LSHTC hybrid classifier had 10 basic classifiers to deal with these 10 subspaces (main topics). The total number of subtopics in the LSHTC dataset was 158. The average number of subtopics to be distinguished by each subspace classifier was therefore 15.8 (158/10). The average vector length handled by a subspace classifier was also 15.8 in this case. The baseline single LSHTC classifiers, however, had to distinguish between all the 158 subtopics and deal with a vector length of 158. The combined effect of a large number of dimensions and a large number of categories considerably reduced the classification accuracies of the baseline single LSHTC classifiers. Thus the gap between the classification accuracies of the hybrid classifier and the baseline single classifiers was very large for the LSHTC dataset. The Reuters datasets, on the other hand, had 4 main topics and 50 subtopics. The Reuters hybrid classifier thus had 4 basic classifiers for the 4 subspaces (main topics). The average number of subtopics to be distinguished by each Reuters subspace classifier was 12.5 (50/4). The average vector length was also 12.5 here. Thus the baseline single Reuters classifiers had to distinguish between 50 subtopics with a vector length of 50. While the complexity to be handled by the subspace classifiers was similar for both LSHTC and Reuters hybrid classifiers (Subtopics and vector lengths of 15.8 v/s 12.5), the complexity to be handled by the baseline single classifiers was very different (Subtopics and vector lengths of 158 for LSHTC v/s 50 for Reuters). The baseline single classifiers for LSHTC thus performed much less well than the baseline single classifiers for Reuters. This was the cause of the greater improvement observed by the LSHTC dataset with the use of hybrid classifiers. Thus increasing the number of level 1 topics (subspaces) causes an increase in the number of subspace classifiers employed by a hybrid classifier thereby causing an increased improvement in the subtopic classification performance.

All these experiments confirmed the facts that:

- The maximum significance value was very effective in detecting the relevant subspace of a test document.
- Training separate classifiers on separate subsets of the original data enhanced overall classification accuracy.
- Hybrid parallel combinations of classifiers trained on different semantic subspaces offered a significant performance improvement over single classifier learning on full data space.
- The use conditional significance vectors increased subtopic classification accuracy.
- Individual classifiers performed better when presented with less data in fewer dimensions.
- The performances of the various hybrid classifiers were very close to each other but all of them performed much better than the baseline single classifiers.
- Datasets with a larger number of categories benefited more from this architecture. This result is particularly encouraging for real-world

applications where the number of categories would be much larger than number present in the experimental datasets.

- Reuters Headlines performed better than Reuters Full Text for the purpose of news classification.

The last finding was consistent across all types of experiments – single classifiers, hybrid 2-classifier combinations as well as hybrid 4-classifier combinations. This can be attributed to the fact that Reuters Full Text contains a lot of text which is introduced to make reading interesting. In the example in Figure 5.7, the body text contains many sentences like “The fast growing company whose outlets are festooned with kitsch movie memorabilia ….” which contain no keywords related to the news topic. From a text processing point of view, this acts as noise which interferes with the relevant words. On the other hand, Reuters Headlines provide a concise summary of the news article which improves classification accuracy.

Chapter 6

Parallel Classifiers - A Special Case: Experiments & Results

In this chapter we look at a special case of the Hybrid Parallel Classifier which we call the Parallel Classifier. This is a meta-classifier which can be implemented with any base classifier. In this architecture, different classifiers *of the same type* are trained on different semantic subspaces using the Conditional Significance Vector with reduced dimensions. In Chapter 5, we combined a classifier of one type such as Bayesian with classifiers of other types such as tree-based, rule-based, etc. Our experiments in Chapter 5 showed that the use of different classifiers on different subspaces improved classification accuracy. However the timing efficiency of such hybrid combinations was not always better than that of all the constituent single classifiers. This was because there was a very wide variation in training times of different types of basic classifiers used. For example, the training time of Naïve Bayes was less than that of MLP by several orders of magnitude. The training time of a Naïve Bayes/MLP combination was between that of single Naïve Bayes and single MLP. In order to focus on timing efficiency, we introduce the Parallel Classifier where different classifiers *of the same type* are used on different data subspaces. This architecture represents a generalised framework with a base classifier. This base classifier can be changed to create different Parallel Classifiers. We experiment with Parallel Classifiers using six different base classifiers. We expect the parallel classifier timings to be of a similar order of magnitude as the corresponding base classifier. The relevant subspace of a document is again detected using the Maximum Significance Value. The datasets used for these experiments are the Reuters Headlines, Reuters Full Text and LSHTC datasets.

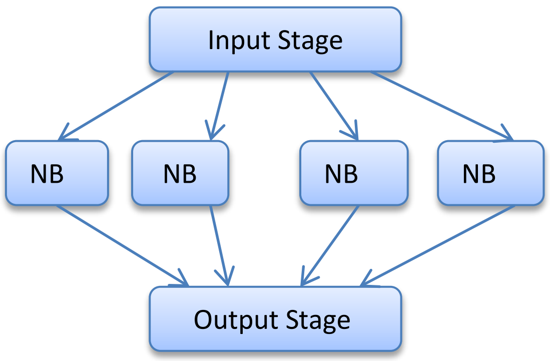
6.1 Parallel Classifier Experiments

These experiments were carried out using six different classification algorithms as the base classifiers for our Parallel Classifier framework. These included two Bayesian algorithms (BayesNet & Naive Bayes), two tree-based algorithms (J48 & Random Forest), one rule-based classifier (PART) and one neural network (MLP). These classifiers were selected to represent a broad range of classification algorithms. The experiments were run using these six algorithms from Weka on the Reuters Headlines and LSHTC datasets. The Conditional Significance Vector representation was used here. The experimental methodology is as follows:

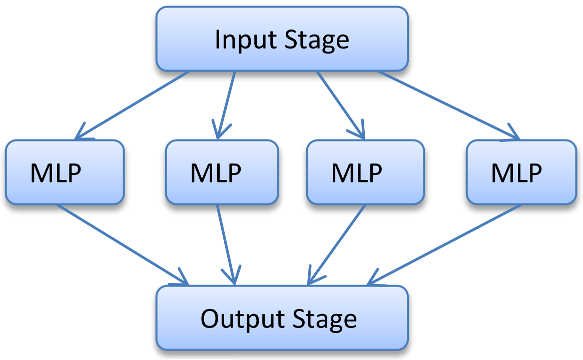
- The Reuters Headlines dataset consisting of 10,000 data vectors, was divided into 9000 training vectors and 1000 test vectors while the LSHTC dataset which had 4463 data vectors was divided into 4000 training and 463 test vectors.

- The 9000 training vectors for Reuters Headlines and the 4000 training vectors for LSHTC dataset were further divided into different subsets according to their main topics.
- These training subsets were then used to separately train the classifiers for each main topic (subspace).
- The classifiers used for different subspaces were all *of the same type.* The figures below show some examples of the Parallel Classifier implementation for the Reuters Corpus. The LSHTC Corpus implementations contain 10 classifiers corresponding to its 10 main topics.

Parallel NB Classifier



PARALLEL CLASSIFIER IMPLEMENTATIONS FOR THE REUTERS CORPUS



Parallel MLP Classifier

- For each subspace, only the vector dimensions representing the subtopics of that main topic were extracted from the complete document vector.
- For a test vector, the main topic was identified by the Maximum Significance Value. Here, all the main topic vector entries were inspected and the maximum among these values was determined. The main topic corresponding to this maximum value was taken as the main topic of the test document.
- The subtopic vector entries corresponding to this predicted main topic were extracted along with their *actual* subtopic label and given to the classifier trained for this main topic (subspace).
- Parallel Classifier architecture was tested using six different base classifiers (Naïve Bayes, BayesNet, J48, Random Forest, PART and MLP).
- In the Reuters parallel classifier using Naïve Bayes, four different Naïve Bayes classifiers were trained on the four subspaces of the Reuters Corpus namely CCAT, ECAT, GCAT and MCAT. Similarly for the Reuters parallel classifier using MLPs, four different MLP classifiers were trained on the four subspaces of the Reuters Corpus and so on.
- As the LSHTC Corpus had ten main topics, the LSHTC parallel classifier using Naïve Bayes had ten basic Naïve Bayes classifiers. These ten basic Naïve Bayes classifiers were trained on the ten subspaces of the LSHTC corpus (Main topics A - J). Similarly for the LSHTC parallel classifier using MLP, ten different MLPs were trained on the ten subspaces (main topics).
- The performance of each single classifier on the full data was compared with the performance of the parallel classifier combination in which this particular classifier was used as a base classifier.
- For the baseline single classifier experiments, the Full Significance Vector and the tf-idf vector representations were used whereas for

the parallel classifier experiments, the category-wise separated Conditional Significance Vector representation was used.

- Parallel Classifier experiments were also run with the Reuters Full Text dataset containing 10,000 items to compare its performance with that of the Reuters Headlines dataset.
- An average of ten runs with different parameter values was taken for all experiments.

6.2 Reuters Corpus Results

The first set of experiments was performed on a dataset of 10,000 headlines drawn from the Reuters Corpus. This dataset had 4 main topics and 50 subtopics.

6.2.1 Reuters Headlines Performance Metrics

In these experiments, we implemented different parallel classifiers with different base classifiers and studied the effect on classification accuracy, training time and test time with the use of this parallel architecture. These experiments were run with 10,000 Reuters Headlines. In all comparisons using this dataset, it was observed that the parallel classifier combination performed better than the single basic classifier. The classification accuracy was improved and the training times as well as classification (test) times were reduced. The baseline using Full Significance Vectors (FSV) performed better than the baseline using tf-idf. Figure 6.1 shows the subtopic classification accuracy, training time and test time for the parallel classifiers along with the baselines for the Reuters Headlines dataset. Figure 6.1(a) shows that the maximum improvement in subtopic classification accuracy was achieved by the Naïve Bayes Classifier while the other classifiers also showed a substantial improvement especially with respect to the tf-idf baseline. Figures 6.1(b) and 6.1(c) show a sharp reduction in training and test times for all classifiers. These figures are shown on a log scale to accommodate a wide range of values. These results are statistically significant for classification accuracy (Friedman test, p=0.0025), training time (Friedman test, p=0.0025) as well as test time (Friedman test, p=0.0057)

Figure 6.2 shows the speed-up of the parallel classifiers with respect to both baselines. Speed-up was calculated by dividing the baseline time by the corresponding parallel classifier time. The speed-up diagrams in Figure 6.2 are also shown on a log scale to accommodate a wide range of values. The maximum training speed-up was achieved by the rule-based classifier PART (14.41 with reference to the FSV baseline and 149.05 with reference to the tf-idf baseline) which was followed by the tree-based classifier J48 (C4.5) with a speed-up of 11.77 with reference to the FSV baseline and 79.50 with reference to the tf-idf baseline.

The test time speed-up was greatest for the Bayesian classifiers. Naïve Bayes achieved a speed-up of 6.08 with respect to FSV and 32.82 with respect to tf-idf, while BayesNet achieved a speed-up of 11.75 and 48.75 with the corresponding baselines. Naïve Bayes achieved significant speed-up in both training and as well as testing (a Train/Test speed-up of 5.84/6.08 and 15.13/32.82 for FSV and tf-idf respectively).

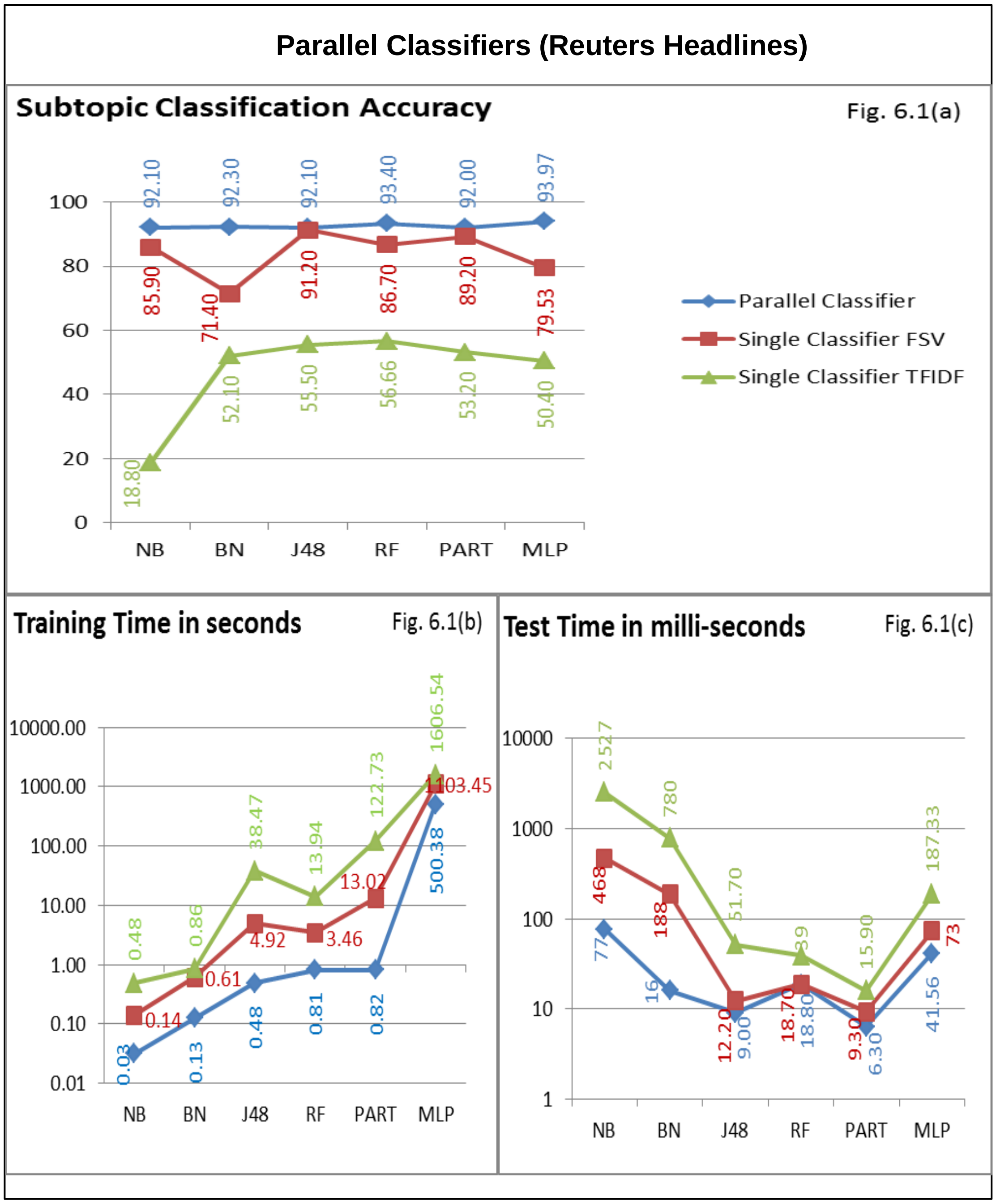


Fig 6.1: Parallel Classifier Performance Metrics with Baselines (Reuters Headlines)

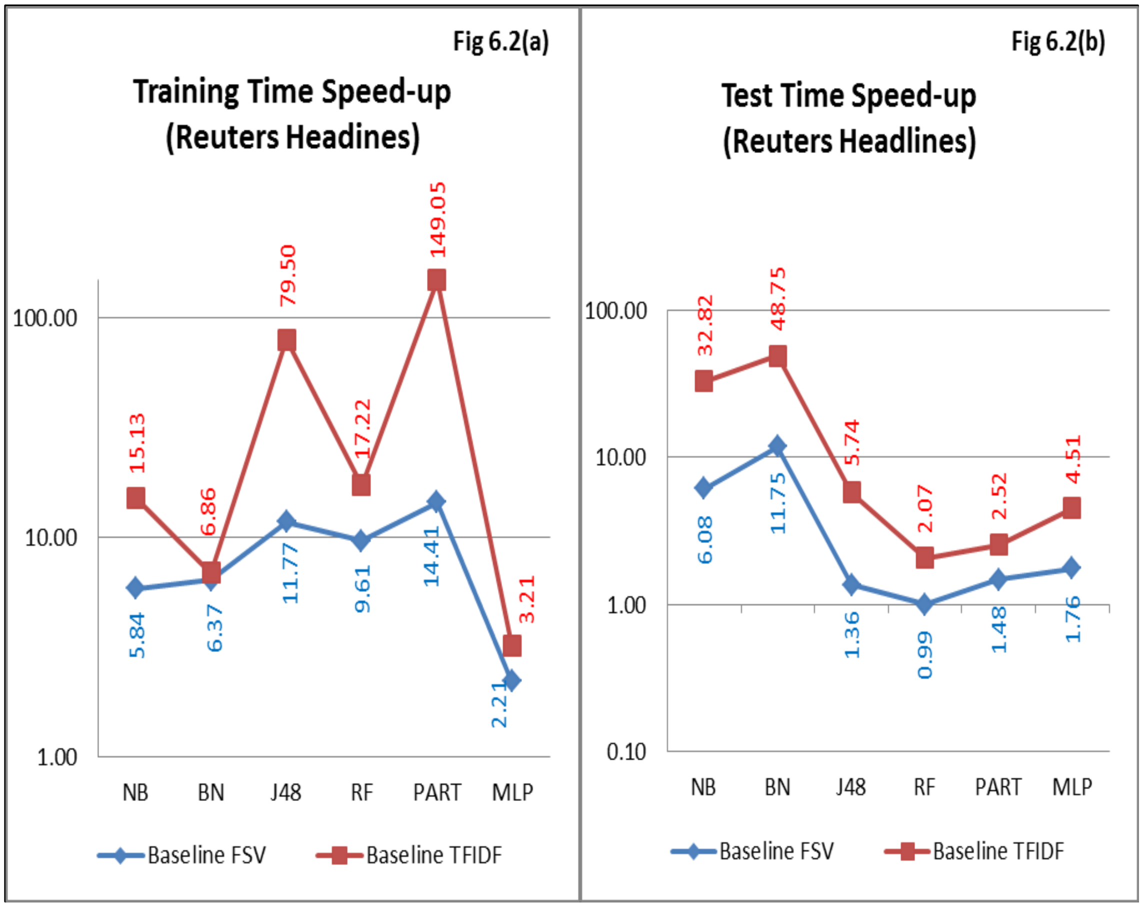


Fig 6.2: Parallel Classifier Speed-up for Reuters Headlines

6.2.2 Comparison of Different Vector Formats for Reuters Headlines

To study the effect of vector format on the Reuters parallel classifier, we also experimented with three different vector formats. These were as follows:

FSV_FullVector- Full Significance Vector with all 50 vector components

FSV_RelVector- Full Significance Vector with only relevant vector components

CSV_RelVector – Conditional Significance Vector with only relevant vector Components

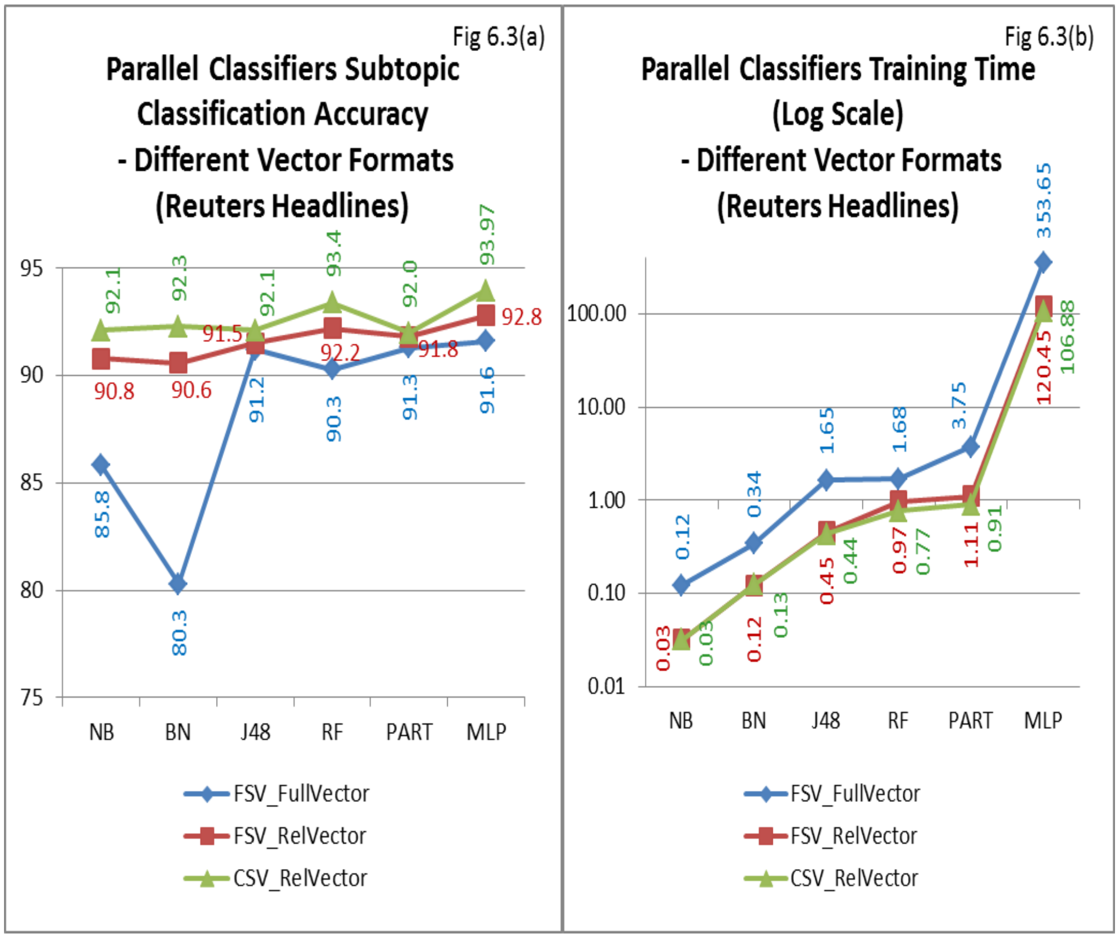
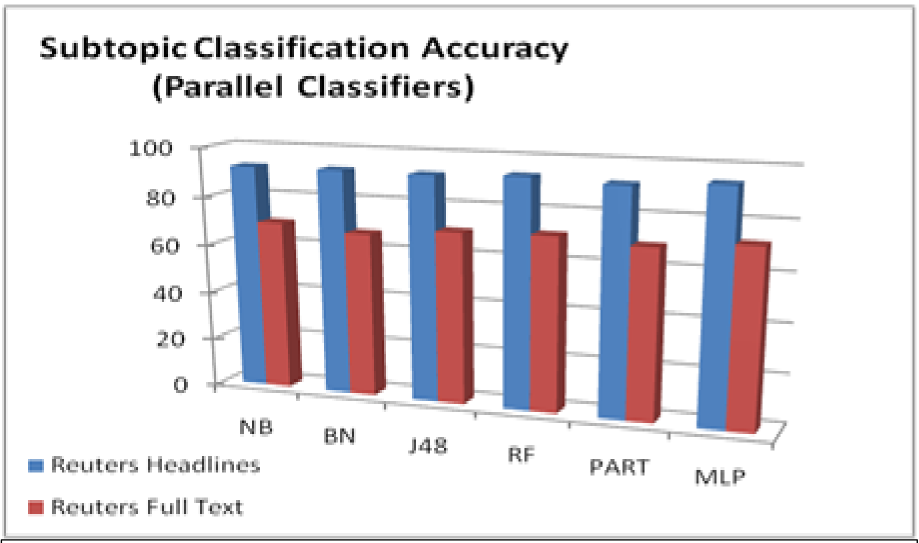


Fig 6.3: Parallel Classifier Metrics with Different Vector Formats (REUTERS)

Figure 6.3 shows the results of these experiments for various parallel classifiers. Figure 6.3(a) shows that the maximum classification accuracy is obtained by the CSV_RelVector for all parallel classifiers. The maximum improvement in classification accuracy is observed in the Bayesian classifiers (Naïve Bayes and BayesNet). The training times in Figure 6.3(b) are again shown on a log scale. A substantial reduction in training times is observed for all classifiers with a reduction in the number of vector components from FullVector to RelVector. The training times of FSV_RelVector and CSV_RelVector are similar for three classifiers – Naïve Bayes, BayesNet and J48. For the other three classifiers – Random Forest, PART and MLP, the CSV_RelVector has the least training time. Thus we observe that a reduction in the number of vector components has a very high impact on the training times of the parallel classifiers. These results are also statistically significant for classification accuracy (Friedman Test, p = 0.0025) as well as training time (Friedman Test, p = 0.0062).

6.2.3 Comparison of Reuters Headlines with Reuters Full Text

The parallel classifier experiments were also run on 10,000 Reuters Full Text news items (containing headlines and body text). It was observed that the subtopic classification accuracy of Reuters news items was better with Reuters Headlines than with Reuters Full Text. This finding was consistent across all parallel classifiers. A possible explanation for this can be that the extra text present in Reuters Full Text acts as noise which degrades classifier



Parallel Classifier Index:

NB – Naïve Bayes BN – BayesNet
J48 – C4.5 Tree RF – Random Forest
PART – Rule Based Classifier MLP – Multilayer Perceptron

Fig 6.4: Comparison of Reuters Headlines and Reuters Full Text

performances. Fig 6.4 shows the corresponding subtopic classification accuracies. This result was statistically significant (Wilcoxon Signed Rank test, p=0.031).

6.3 LSHTC Corpus Results

In order to test the effect of a large number of categories on the Parallel Classifier, we also ran these experiments with the Large Scale Hierarchical Text Collection (LSHTC) which had 10 main topics and 158 subtopics.

6.3.1 LSHTC Performance Metrics

Figure 6.5 below shows the performance of six different parallel classifiers along with the baselines for the LSHTC corpus. It was observed that the parallel classifier combinations performed better than all the corresponding single basic classifiers. In this case too, the classification accuracy was improved and the training times as well as classification (test) times were reduced. Figure 6.5 shows the subtopic classification accuracy, training time and test time for the parallel classifiers along with the baselines for the LSHTC dataset. Figure 6.5(a) shows a substantial improvement in the classification accuracy of all parallel classifiers with respect to both baselines. The performance of the FSV baseline was better than the performance of the tf-idf baseline. Figures 6.5(b) and 6.5(c) show a sharp reduction in training and test times for all classifiers. These figures are shown on a log scale to accommodate a wide range of values. In this case, the performances of the two baselines, Full Significance Vector (FSV) and tf-idf, were similar to each other. A possible explanation for this could be the similar vector length of these two baselines. These results are again statistically significant for classification accuracy (Friedman test, p=0.0025), training time (Friedman test, p=0.0111) as well as test time (Friedman test, p=0.0111).

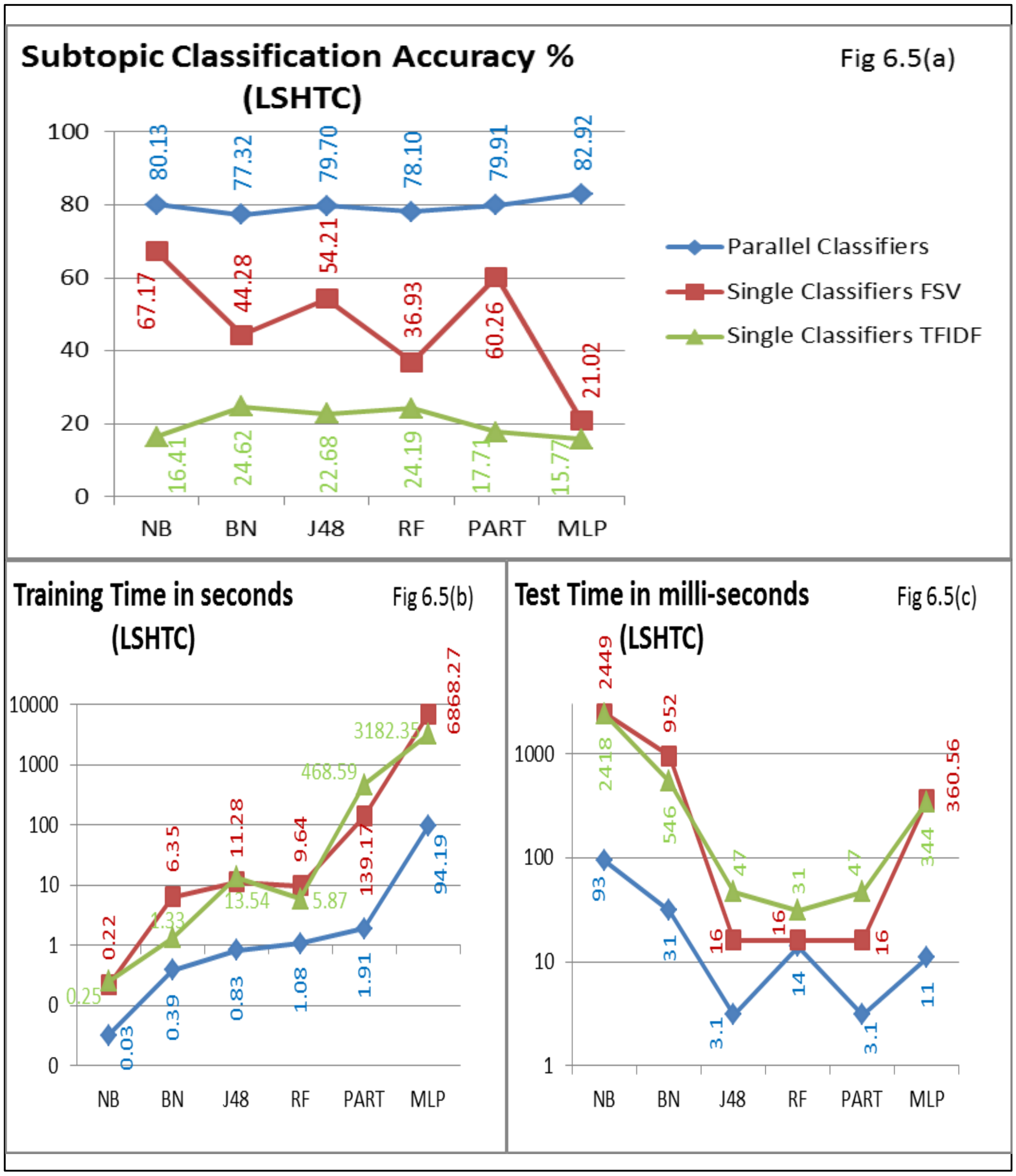


Fig 6.5: Parallel Classifier Performance Metrics with Baselines (LSHTC)

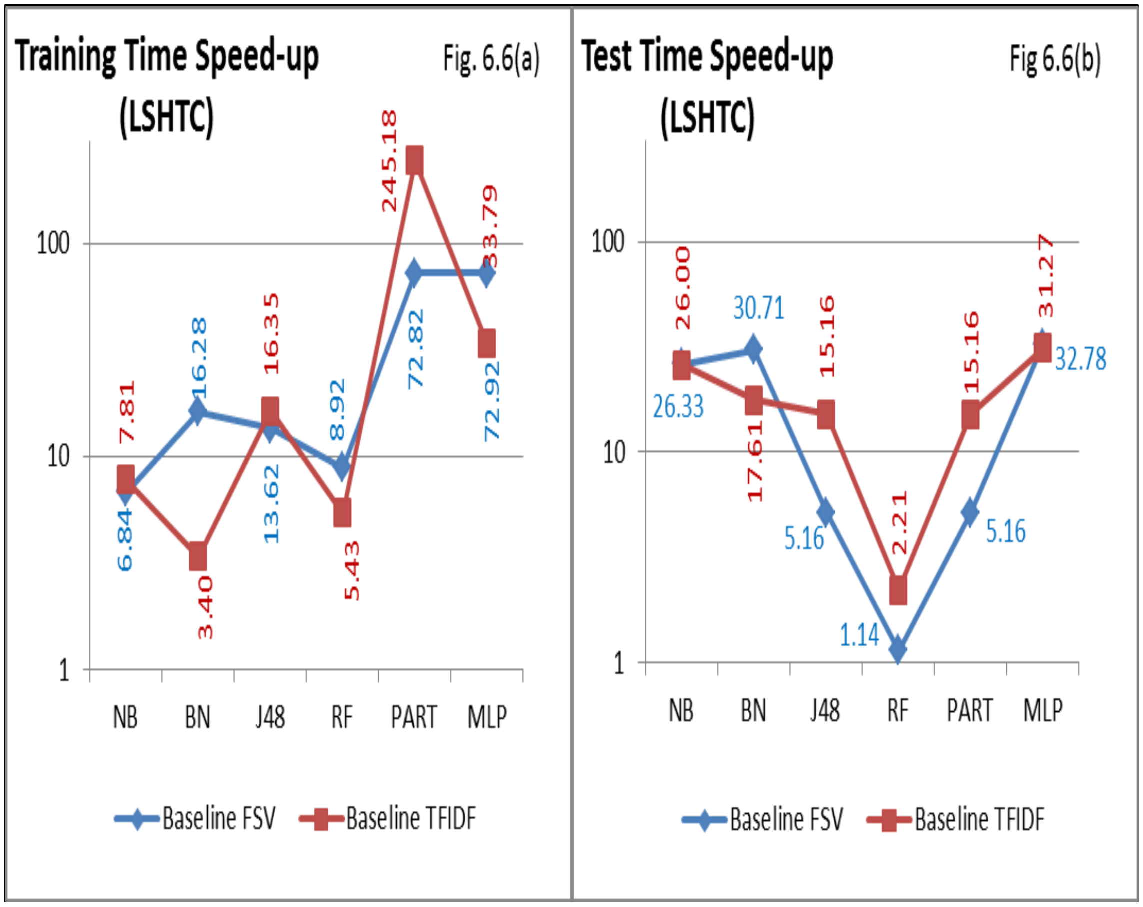


Fig 6.6: Parallel Classifier Speed-up for LSHTC

Figure 6.6 shows the speed-up of the parallel classifiers with respect to both baselines. The speed-up diagrams in Figure 6.6 are also shown on a log scale to accommodate a wide range of values. Similar to the Reuters dataset, the maximum training speed-up was achieved by the rule-based classifier PART (72.82 with reference to the FSV baseline and 245.18 with reference to the tf-idf baseline). This was followed by MLP with a speedup of 72.92 for FSV and 33.79 for tf-idf. The maximum test speedup was achieved by MLP (32.78/31.27 for FSV/tf-idf) followed by Naïve Bayes (26.33/26.00 for FSV/tf-idf). All the classifiers showed a good speedup for both training and test times (except Random Forest). For classification (test) times, Random Forest has a very low speedup with reference to the FSV baseline - 0.99 with Reuters and 1.14 with LSHTC. The reason for this could be that Random Forest proceeds by creating a number of trees (default 10 trees in Weka). These trees are created during the training phase. In a parallel classifier, each Random Forest algorithm works with a reduced subset of data and therefore the time required to construct these trees (training time) is reduced. However, the *number* of trees created remains the same as the number created with the baseline single Random Forest classifier. The test vector has to proceed down 10 trees and voting is done among 10 labels in both cases. Hence the test timings remain similar and therefore the test speed-up is low.

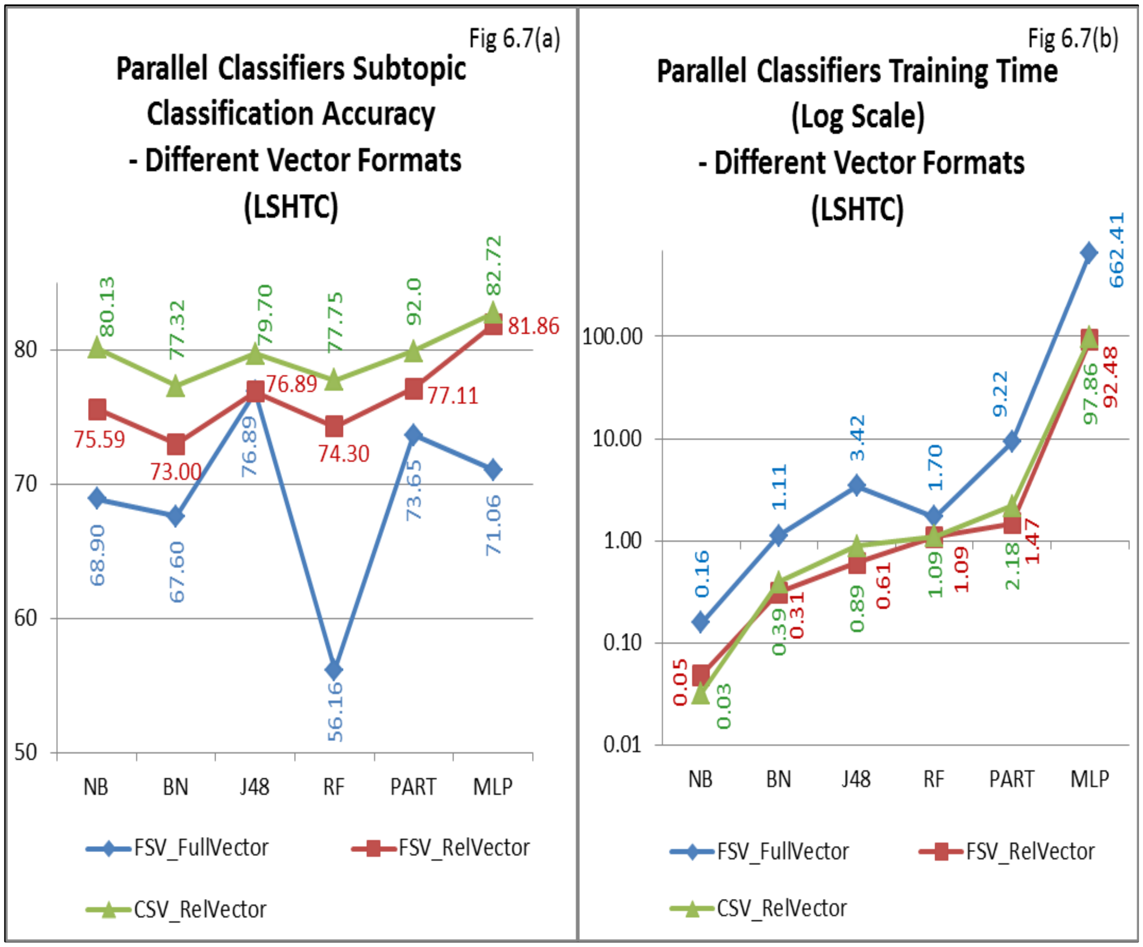


Fig 6.7: Parallel Classifier Metrics with Different Vector Formats (LSHTC)

6.3.2 Comparison of Different Vector Formats for LSHTC

Figure 6.7 shows the LSHTC parallel classifier performance metrics for different vector formats. These vector formats were the FSV_FullVector, the FSV_RelVector and the CSV_RelVector. We can see that the CSV_RelVector gave the highest classification accuracy and all classifiers except Random Forest showed a substantial decrease in training time in going from FullVector to RelVector. Thus a reduction in the number of vector components leads to a big reduction in training time in this case too. The time axis is shown on a log scale here. The performances of the FSV_RelVector and CSV_RelVector were similar. These results are again statistically significant for classification accuracy (Friedman Test, p = 0.0031) as well as training time (Friedman Test, p = 0.0057)

6.4 Conclusion

The results in this chapter show that:

- The Maximum Significance Value is very effective in detecting the relevant subspace of a test document.
- Combining classifiers of the *same type* in parallel improves the classification accuracy of the concerned basic classifier where the underlying data has distinct semantic categories.
- Reuters Headlines perform better than Reuters Full Text for the purpose of news classification.
- A parallel combination of classifiers results in a very sharp reduction in training and testing times. The speed-up achieved is considerable in all cases.

All of these results are statistically significant. Naïve Bayes achieved a high degree of speed-up in *both* training and test timings along with the greatest improvement in classification accuracy. Since Naïve Bayes is already a fast classifier, further speed-up can be put to good use especially in search technology. The results showed that Naïve Bayes achieved a train/test speed up of 5.84/6.08 and 15.13/32.82 for FSV and tf-idf respectively for the Reuters dataset. The corresponding values for the LSHTC dataset were 6.84/26.33 and 7.81/26.00. Thus parallel classifiers work well even with a larger hierarchy (4 main topics and 50 subtopics for Reuters vs. 10 main topics and 158 subtopics for LSHTC). This is an encouraging result as real world hierarchies are much larger than the experimental ones.

Chapter 7

Conclusion

7.1 Introduction

The text documents available to us nowadays are often related to a taxonomy or category hierarchy rather than a flat classification system. The documents present in today's world are also increasing exponentially. To be able to access these documents in real time, we need fast automatic methods which take advantage of these category hierarchies. As the documents become very diverse it becomes difficult for a single classifier to deal with such varied data. In this thesis, we looked at combinations of diverse classifiers to improve classification of text documents in a two-level hierarchy.

7.2 Achievement of Objectives

The following objectives mentioned in section 1.2.2 have been achieved:

1. *Conduct a literature review on the current state of multilevel text classification systems using machine learning methods with emphasis on hierarchical classification and subspace learning*

In chapter 2, a detailed literature survey and analysis of the current state of art techniques is presented for both hierarchical classification (section 2.1.3) and subspace learning (section 2.2). This study clarified that it is appropriate to concentrate on at most 2 – 3 levels of category hierarchy to avoid the problem of error propagation. It also showed that the use of subspace learning can deal with a two-level hierarchy while at the same time increasing the speed of subspace detection by the use of search methods rather than classification techniques at level 1 of the hierarchy.

2. *Propose a new vector representation suitable for two-level learning*

Chapter 3 looks at an existing method of category based vector representation. This method weighs different words according to their importance in different categories. We proposed a modification to this method to incorporate a two-level hierarchy and weigh words according to their importance within a particular data subspace (main topic). Our new vector representation can be expanded to incorporate a multi-level hierarchy.

3. *Conduct a literature review on the currently available methods of classifier combination.*

4. *Research various classifier combination methods to improve two-level text classification.*

Section 2.3 of this thesis showed that most classifier combination methods concentrated on a flat classification system with few categories. They concentrated on feature transformation techniques to reduce the feature set and then applied ensemble methods for classification. Subspace Clustering, which works by extracting a data subspace along with features relevant to that data subspace was more appropriate for our problem domain of text with a large number of categories. However, even this method had not specifically been applied to two-level classification. We detected a gap in the literature where category information could be used to identify subspaces instead of the use of unsupervised clustering.

5. *Propose a new hybrid architecture for improved two-level learning using the new proposed vector representation*.

In Chapter 4, we presented a new hybrid architecture based on subspace detection using a novel method based on the Maximum Significance Value extracted from a single document vector and the use of separate classifiers for separate subspaces to improve level 2 learning.

6. *Evaluate the performance of the new proposed hybrid architecture using various performance methods*

Chapter 5 presents evaluation of this architecture using the metrics of classification accuracy and training time. The standard measures of precision and recall are shown to be equivalent to classification accuracy for our case of single label multi-class classification. The results of statistical significance tests are also presented here. Chapter 6 presents the evaluation of a special case of this architecture – the Parallel Classifier, on classification accuracy, training time as well as test time. Statistical significance testing is also presented for this case.

7.3 Review of Hypothesis and Research Questions

7.3.1 Hypothesis

The use of separate classifiers for separate subspaces will improve overall subspace classification accuracy and learning time and lead to improved two-level classification of text documents.

Our experiments in Chapter 5 have proved this hypothesis to be correct. In these experiments, the experimental datasets were divided into different subspaces or subsets based on the level 1 (main) category.

The Experiment Set A (section 5.1) combined MLP in parallel with other classifiers in various hybrid two-classifier and four-classifier combinations where each classifier operated on a separate data subspace. These experiments were conducted on the Reuters Headlines dataset and the LSHTC dataset. For a two-classifier combination, MLP and the other classifier were used alternately on the main category topics while for a four-classifier system four different classifiers were used on the four main topics of the Reuters Headlines dataset and repeated for each block of four main topics for the LSHTC dataset. The baseline for these experiments was a single MLP classifier on the full data dataset using two different vector formats – the tf-idf and the Full Significance Vector. The results of these experiments (section 5.1.2) showed that the use of hybrid MLP-based classifiers resulted in an improvement in subtopic classification accuracy along with a significant reduction in training time over the single MLP baselines for both the datasets.

In Experiment Set B (section 5.2), various hybrid 2-classifier and 4-classifier combinations were tested. For the hybrid 2-classifier combinations, a classifier of one type was combined with classifiers of other types in a large variety of combinations. The performance of each single classifier on the full data was compared with the performance of the hybrid 2-classifier combinations in which this particular classifier also participated. For the baseline single classifier experiments, the Full Significance Vector representation was used, whereas for the hybrid classifier experiments, the category-wise separated Conditional Significance Vector representation was used. In these experiments, in addition to Reuters Headlines and LSHTC, the Reuters Full Text dataset was also used. Reuters Full Text merged headlines with the body text for each Reuters news item. The results for these experiments (sections 5.2.1 & 5.2.2) showed that classification accuracy of all the hybrid two-classifier combinations were better than that of the corresponding baseline single classifiers and the hybrid four-classifier combinations performed better than all of the single classifiers.

7.3.2 Research Questions

The three research questions given in Chapter 1 are as follows:

Research Question 1) *Is it possible to devise a method to quickly direct the document search to a relevant document subspace by examining only a single input query vector?*

In the initial experiments presented in section 4.1.1 with 10,000 Reuters Headlines, the subspace of a test document was detected by the Maximum Significance Value which is the maximum value among the level 1 topic entries of the Conditional Significance Vector. The categories within this subspace were then allocated by a single classifier. The performance of ten different classifiers was compared for this process. All classifiers showed that the best subtopic classification accuracy was achieved by keeping only the relevant subspace in consideration. Thus the correct subspace is detected in this case by examining only a single input query vector. Further experiments conducted with a larger dataset (section 4.1.2 – 100,000 Reuters Headlines) and with longer documents (section 4.1.3 – 10,000 Reuters Full Text items) and with a dataset with a larger hierarchy (section 4.2 - LSHTC) all showed that the performance of level 2 classification is improved with basic classifiers when only the relevant subspace detected by our method is used. Hence it is possible to devise a method to quickly direct the document search to a relevant document subspace by examining only a single input query vector.

Research Question 2) *Can we develop a classification method which directs the classification from all possible classes to a relevant subspace of classes?*

The developed method is as follows: We take an existing category based vector representation system, the Significance Vector (section 3.1.2) and modify it to represent a two-level hierarchy with the Conditional Significance Vector (section 3.1.3). In the Conditional Significance Vector, the initial vector components (4 for Reuters / 10 for LSHTC) contain information about level 1 while the remaining components contain information about level 2. We use the Maximum Significance Value among the level 1 components to detect the level 1 (main) category of a document. After this, only the vector components relevant to the subtopics of that main topic are extracted and given to a classifier for subtopic classification. This directs the classification from all possible classes to a relevant subspace of classes. The improved subtopic classification accuracies obtained by using this method in chapters 4, 5 and 6 show the strength of our method.

Research Question 3) *Is it possible to have a vector representation that focuses on the relative importance of keywords within a data subspace?*

The word significance vectors discussed in section 3.1.2 take the relative weight of a word for all categories in the system. At subtopic level, this means taking all the subtopics in the dataset under consideration. This forms the Full Significance Vector. In section 3.1.3, we have proposed the Conditional Significance Vector which takes the relative weight of a keyword with reference to their relevant data subspace. At subtopic level, this means all the subtopics of only a particular main topic – not all the subtopics in the complete dataset. A keyword which is more important in subspace A than in subspace B receives a higher weightage in subspace A than in subspace B. Thus the weight of the same word is different in different subspaces. The results of our experiments comparing different vector formats (Fig 5.6, Fig 6.3 and Fig 6.7) show that the best subtopic classification accuracy is obtained by the Conditional Significance Vector Hence it is possible to have a vector representation that focuses on the relative importance of keywords within a data subspace.

7.4 Summary of Contributions

This work contributes to the research on text classification in the presence of multi-level category hierarchies and to the research on hybrid methods of classifier combination. The main contributions are as follows:

- Identifying the need for a fast classification method which can deal with category hierarchies.
- Proposal of a new vector format, the Conditional Significance Vector which encodes category hierarchy information along with the relative importance of words in different data subspaces.
- Proposal of a new method of subspace detection using the Maximum Significance Value extracted from the Conditional Significance Vector.

- The first application of subspace learning using category information to deal with two-level text data.
- Proposal of a Hybrid Parallel Classifier to improve learning at the sub-category level.
- Conducting a variety of experiments with a large number of hybrid combinations to show the efficacy of this method.
- Experiments with a special case, the Parallel Classifier, to show improvement in a single type of classifier for two-level text classification using this high level architecture.

The main findings of this work were as follows:

- Maximum Significance Value is very effective in identifying the relevant subspace of a test document. It is also very fast as it examines just a single document vector for subspace detection.
- The use of Conditional Significance Vectors enhances the distinction between subtopics in a subspace improving overall learning at level 2.
- The use of separate classifiers for separate subspaces improves learning at the subtopic level.
- There is very small variation in the performance of different hybrid and parallel combinations but all of them perform much better than the single baseline classifiers.
- The method of classifier combination is more important than the classifiers themselves. Our architecture can thus be implemented with any base classifier available.
- The improvement obtained for level 2 learning with the Hybrid Parallel Classifiers increases as the category hierarchies become larger. This is shown by the higher improvement obtained by the LSHTC Corpus with 10 main and 158 subtopics than the Reuters Corpus with 4 main and 50 subtopics.

- The use of the same type of classifier for different subspaces results in a considerable reduction in training and test timings along with an improvement in the classification accuracy.
- Reuters Headlines perform better than Reuters Full Text (Headlines + Body Text) for the purpose of news classification with our vector representation.

7.5 Applicability to Other Domains

In this work, we have applied our techniques to unstructured text. However, Hybrid Parallel Classifiers can be applied to many other domains such as Image Processing, Pattern Recognition and Computer Vision where different classifiers can work on different parts of an image/pattern to improve overall recognition. The image vector can be constituted such that different blocks of the image vector correspond to distinct parts of an image. In face recognition, different classifiers can be allocated to recognise different parts of a face such as eyes, ears, mouth and nose. The outputs of these different classifiers can then be combined to generate a final face recognition decision. Apart from image data, this technique can also be applied to image captions for image classification. Computational Biology can also benefit from this method to improve recognition within subdomains. Social Media, which has a lot of text content, can also be explored with this method for customised suggestions and marketing.

7.6 Personal Concluding Remarks

This research has greatly deepened my understanding of the challenges faced by the field of unstructured text classification in the presence of the ever increasing volume and complexity of data and category hierarchies. It has also increased my understanding of machine learning methods and their importance in today's world. I am particularly satisfied that my instinct of pointing directly to a subset of data for improving subtopic classification could be practically implemented and has shown encouraging results. I am, however, aware that category hierarchies are not static in today's world. As such this system needs to be made dynamic.

Our experimental results show that our high level architecture is very good in improving learning at level 2. This improvement is almost independent of the types of classifiers used. Even though there is variation in the performance of various hybrid and parallel classifiers, they are very close to each other and all of them are much better than the baseline single classifiers over the full data space. This seems to suggest that it is our general architecture of maximum significance based subspace learning which improves performance. Thus this architecture can be implemented with any base classifiers available. An elementary classifier such as Naïve Bayes performs as well with this architecture as a more complex classifier such as the MLP. The strength of our architecture thus lies in the method of classifier combination rather than the classifiers themselves. Thus it can act as a powerful meta-classifier whose performance is even more useful when the underlying data has a more complex category hierarchy. This has been shown by a higher improvement observed over the baselines in the LSHTC Corpus with 10 main and 158 subtopics than in the Reuters Corpus with 4 main and 50 subtopics.

A major implication for this in the field of text classification and classifier theory is that further improvements in classification performances can be obtained by combining various classifiers and by focusing on improving feature vector representation. Vector representations that incorporate semantic information give better results than general vectors like tf-idf. Thus feature engineering and classifier combinations should be the focus of future work for improved classification performances.

7.7 Suggestions for Further Work

a) Use of Ontologies:

In this work, significance vectors are calculated solely on the basis of word frequencies. The addition of domain ontologies to the computation of Conditional Significance Vectors should further improve learning.

b) Error Correction:

To keep classification speeds high, we have avoided error correction in this work. Gao et al. (2009) have applied error reduction and correction in a hierarchical classification scheme using category and subtree probabilities. Similar methods of error correction can be explored to increase the classification accuracy of our system.

c) Use of classifier combinations:

Classifier ensembles can be used instead of single classifiers to process each subspace. Classifier ensembles are based on the reasoning that strengths and weaknesses of various classifiers can compensate each other. An instance which is misclassified by one classifier may be classified correctly by another classifier thus pushing up the combined classification performance. Classifier ensembles have been successfully applied in the literature to text with flat classification schemes [(Larkey and Croft (1996), Al-Kofahi et al. (2001), Florian et al. (2003), Fradkin and Kantor (2005)]. In our Hybrid Parallel Classifier, the classifiers operating in the subspaces deal with a single level of categories (only the level 2 subcategories within a main category). Hence classifier ensembles should improve learning within a subspace.

d) Extension to more levels:

We have applied the Maximum Significance Value to detect the main topic at level 1. This method can be applied recursively at lower levels to deal with a deeper hierarchy. For example, in a three level hierarchy, the reduced length sub-vector extracted after subspace detection at level 1 will contain information about level 2 as well as level 3. The Maximum Significance Value can be applied to the level 2 category information to further detect a lower level subspace and extract a further reduced vector containing only level 3 information. A classifier can then be applied at level 3 to obtain the level 3 category relevant to the document being processed.

e) Modifications to deal with dynamic category hierarchies:

In today’s world, category hierarchies are continuously changing. New categories are introduced and old ones phased out. Our system is based on a fixed category hierarchy. Methods of converting this to a dynamic architecture should be explored.

Bibliography

Aha, D. W., Kibler, D., & Albert, M. K. (1991). "Instance Based Learning Algorithms". *Machine Learning, 6*, 37-66.

Al-Ani, A., & Deriche, M. (2002). "A New Technique for Combining Multiple Classifiers using the Dempster-Shafer Theory of Evidence". *Journal of Artificial Intelligence Research, 17*, 333-361.

Al-Kofahi, K., Tyrrell, A., Vachher, A., Travers, T., & Jackson, P. (2001). "Combining multiple classifiers for text categorization". *Proceedings of the tenth international conference on Information and knowledge management , CIKM 2001*, (pp. 97-104).

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., Paliouras, G., & Spyropoulos, C. D. (2000). An Evaluation of Naive Bayesian Anti-Spam Filtering. In G. Potamias, V. Moustakis, & M. van Someren (Ed.), *Proceedings of the workshop on Machine Learning in the New Information Age, 11th European Conference on Machine Learning*, (pp. 9-17). Barcelona, Spain.

Bacan, H., Pandzic, I. S., & Gulija, D. (2005). "Automated News Item Categorization". *Proceedings of the 19th Annual Conference of The Japanese Society for Artificial Intelligence* (pp. 251-256). Springer-Verlag.

Bagan , H., & Yamagata, Y. (2010). "Improved Subspace Classification Method for Multispectral Remote Sensing Image Classification". *Photogrammetric Engineering & Remote Sensing, 76*(11), 1239-1251.

Basu, A., Watters, C., & Shepherd, M. (2003). "Support Vector Machines for Text Categorization". *Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 4 - Volume 4*, (p. 103.3).

Bernard, S., Heutte, L., & Adam, S. (2009). “On the selection of decision trees in Random Forests”. *Proceedings of the International Joint Conference on Neural Networks (IJCNN '09)*, (pp. 790-795).

Boutemedjet, S., Ziou, D., & Bouguila, N. (2010). "Model-based subspace clustering of non-Gaussian data". *Neurocomputing*, 1730-1739.

Breiman, L. (1996). “Bagging predictors”. *Machine Learning, Vol 24*(Issue 2), pp 123-140.

Breiman, L. (2001, October). “Random Forests”. *Machine Learning, Vol 45*(Issue 1, October 2001), pp 5-32.

Cai, L., & Hofmann, T. (2004). "Hierarchical Document Categorization with Support Vector Machines". *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM'04)*, (pp. 78-87).

Calabuig, D., Giminez, S., Roman, J., & Monserrat, J. (2010). "Fast Hopfield Neural Networks using Subspace Projections". *Neurocomputing, 73*, 1794-1800.

Cao, B., Shen, D., Sun, J.-T., Yang, Q., & Chen, Z. (2007). "Feature Selection in a Kernel Space". *Proceedings of the 24th International Conference on Machine Learning.* Corvallis, OR.

Chatpatanasiri, R., & Kijsirikul, B. (2010). "A unified semi-supervised dimensionality reduction framework for manifold learning". *Neurocomputing, 73*, 1631-1640.

Chen, J., Ji, S., Ceran, B., Li, Q., Wu, M., & Ye, J. (2008). "Learning Subspace Kernels for Classification". *The 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*, (pp. 106-114). Las Vegas.

Chen, K., Gao, S., Zhu, Y., & Sun, Q. (2006). "Music Genres Classification Using Text Categorization Method". *IEEE 8th Workshop on Multilmedia Signal Processing*, (pp. 221-224).

Chuang, W. T., Tiyyagura, A., Yang, J., & Giuffrida, G. (2000). "A Fast Algorithm for Hierarchical Text Categorization". *Proceedings of the Second International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2000)*, (pp. 409-418). London.

Cohen , M. C., & Paliwal, K. K. (2008). "Classifying microarray cancer datasets using nearest subspace classification". *Proceedings of the third IAPR International Conference on Pattern Recognition in Bioinformatics (PRIB 2008).* Melbourne.

Duin, R. P. (2002). "The Combining Classifier: to Train or Not to Train". *Proceedings of the 16th IEEE International Conference on Pattern Recognition*, *2*, pp. 765-770. Quebec City, Canada.

Duin, R., & Tax, D. (2000). "Experiments with Classifier Combining Rules". (J. Kittler, & F. Roli, Eds.) *MCS 2000, LNCS 1857*, pp. 16−29.

Dumais, S., Platt, J., Heckerman, D., & Sahami, M. (1998). "Inductive Learning Algorithms and Representations for Text Categorization". *In Proceedings of ACM 7th International Conference on Information and Knowledge Management (CIKM 98)*, (pp. 148-155).

Estabrooks, A., & Japkowicz, N. (2001). "A mixture-of-experts framework for text classification". *Proceedings of the 2001 Workshop on Computational Natural Language Learning*, *7, July 06 - 07, 2001*, pp. 1-8. Toulouse, France.

Florian, R., Ittycheriah, A., Jing, H., & Zhang, T. (2003). "Named Entity Recognition through Classifier Combination". *Proceedings of the seventh conference on Natural language learning (CONLL '03) at HLT-NAACL 2003. 4*, pp. 168-171. Edmonton: Association for Computational Linguistics.

Fradkin, D., & Kantor, P. (2005). "Methods for Learning Classifier Combinations : No Clear Winner". *ACM Symposium on Applied Computing*, (pp. 1038-1043).

Frank, E., & Witten, I. (1998). “Generating Accurate Rule Sets Without Global Optimization”. In J. Shavlik (Ed.), *Machine Learning: Proceedings of the Fifteenth International Conference.* Morgan Kaufmann Publishers.

Frank, E., Wang, Y., Inglis, S., Holmes, G., & Witten, I. H. (1998). "Using model trees for classification". *Machine Learning, 32*(1), 63-76.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). "Additive Logistic Regression: A Statistical View of Boosting". *The Annals of Statistics, 28*(2), 337-407.

Fu, Y., Cao, L., Guo, G., & Huang, T. S. (2008). "Multiple Feature Fusion by Subspace Learning". *Proceedings of the 7th ACM International Conference on Image and Video Retrieval (CIVR’08)* (pp. 127-134). Niagara Falls, Ontario, Canada.: ACM.

Fukumoto, F., & Suzuki, Y. (2002). "Manipulating Large Corpora for Text Classification". *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (pp. 196-203). Philadelphia.

Fukunaga, K. (1990). *"Introduction to Statistical Pattern Recognition"* (2nd ed.). New York: Academic Press.

Gangeh, M. J., Kamel, M. S., & Duin, R. P. (2010). "Random Subspace Method in Text Categorization". *IEEE 20th International Conference on Pattern Recognition (ICPR)*, (pp. 2049-2052). Istanbul.

Gao, F., Fu, W., Zhong, Y., & Zhao, D. (2009). "Large-Scale Hierarchical Text Classification Based on Path Semantic Vector and Prior Information". *International Conference on Computational Intelligence and Security (CIS'09)*, (pp. 54-58). Beijing.

Garcia-Pedrajas, N., & Ortiz-Boyer, D. (2008). "Boosting Random Subspace Method". *Vol 21(2008)*, pp 1344-1362.

Ghazi, D., Inkpen, D., & Szpakowicz, S. (2010). "Hierarchical versus Flat Classification of Emotions in Text". *NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, (pp. 140-146). Los Angeles.

Giorgetti, D., & Sebastiani, F. (2003). "Multiclass text categorization for automated survey coding". *Proceedings of the 18th ACM Symposium on Applied Computing (SAC 2003)*, (pp. 798-802). Melbourne, US.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. (2009, July). "The WEKA Data Mining Software: An Update,". *ACM SIGKDD Explorations Newsletter, Volume 11*(Issue 1), pp. 10-18.

He, X., & Cai, D. (2009). "Active Subspace Learning". *IEEE 12th International Conference on Computer Vision (ICCV)*, (pp. 911-916). Kyoto, Japan.

Hendersen, L. (2009). *"Automated Text Classification in the DMOZ Hierarchy".* Australian National University (ANU). Canberra, Australia: CiteSeer.

Ho, T. K. (1998). "The random subspace method for constructing decision forests". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *Volume 20, Issue 8 (Aug 1998)*, pp. 832-844.

Hotho, A., Staab, S., & Maedche, A. (2001). "Ontology-based Text Clustering". *Proceedings of the IJCAI-2001 Workshop “Text Learning: Beyond Supervision"*, (pp. 30-37). Seattle.

Hu, R., Shen, W., & Wang, H. (2010). "Recursive spatiotemporal subspace learning for gait recognition". *Neurocomputing, 73*, 1892-1899.

Jing, L., Ng, M. K., Xu, J., & Huang, J. Z. (2005). "Subspace Clustering of Text Documents with Feature Weighting K-Means Algorithm". *PAKDD 2005, LNAI 3518* (pp. 802-812). Springer-Verlag.

Joachims, T. (1998). "Text Categorization with Support Vector Machines: Learning with Many Relevant Features". *European Conference on Machine Learning (ECML)* (pp. 137-142). Berlin: Springer.

Joliffe, I. (1986). *"Principle Component Analysis".* New York: Springer-Verlag.

Jordan, M. I., & Jacobs, R. A. (1993). "Hierarchical mixtures of experts and the EM algorithm ". *Proceedings of the 1993 IEEE International Joint Conference on Neural Networks (IJCNN'93)*, *2*, pp. 1339-1344. Nagoya.

Kiang, M. Y. (2003). "A comparative assessment of classification methods". *Decision Support Systems, 35*, 441-454.

Kim, S.-B., Han, K.-S., Rim, H.-C., & Myaeng, S. H. (2006, November). "Some Effective Techniques for Naive Bayes Text Classification". *IEEE Transactions on Knowledge and Data Engineering, 18*(11), 1457-1466.

Kittler, J., Hatef, M., Duin, R. P., & Matas, J. (1998, March). "On Combining Classifiers". *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*(3), 226-239.

Koller, D., & Sahami , M. (1997). "Hierarchically classifying documents using very few words". *Proceedings of the Fourteenth International Conference on Machine Learning (ICML-97)*, (pp. 170-178). Nashville, Tennessee.

Kosmopoulos, A., Gaussier, E., Paliouras, G., & Aseervatham. (2010). "The ECIR 2010 Large Scale Hierarchical Classification Workshop". *SIGIR Forum*, *44, Number 1, June 2010*, pp. 23-32.

Kotsiantis, S. (2009). "Local Random Subspace Method for constructing multiple decision stumps". *International Conference on Information and Financial Engineering, 2009*, (pp. 125-129).

Kotsiantis, S. B. (2007). "Supervised Machine Learning: A Review of Classification Techniques". *Informatica, 31*, 249-268.

Kwak, N., & Lee, J. (2010). "Feature extraction based on subspace methods for regression problems". *Neurocomputing, 73*, 1740-1751.

Larkey, L. (1998). "Automatic Essay Grading Using Text Categorization Techniques". *21st ACM International Conference on Research and Development in Information Retrieval (SIGIR-98)*, (pp. 90-95).

Larkey, L. S., & Croft, W. B. (1996). "Combining Classifiers in Text Categorization". *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'96)*, (pp. 289-297). Zurich.

Law, Y. N., Lee, H. K., & Yip, A. M. (2010). "Semi-Supervised Subspace Learning for Mumford-Shah Model Based Texture Segmentation". *Optics Express 4434 (Optical Society of America), 18*(5), 4434-4448.

Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). "RCV1: A New Benchmark Collection for Text Categorization Research". *Journal of Machine Learning Research, 5*, 361-397.

Li , Y. H., & Jain, A. K. (1998). "Classification of Text Documents". *The Computer Journal, 41*(8), 537-546.

Li, T., Zhu, S., & Ogihara, M. (2003). "Efficient Multi-way Text Categorization via Generalized Discriminant Analysis". *Proceedings of the twelfth international conference on Information and knowledge management (CIKM '03)* (pp. 317-324). New Orleans, USA: ACM.

Li, Y. (2004). "On incremental and robust subspace learning". *Pattern Recognition, 37*, 1509-1518.

Likforman-Sulem, L., & Sigelle, M. (2008, October). “Recognition of degraded characters using dynamic Bayesian networks”. *Pattern Recognition, Vol 41*(Issue 10, October 2008), pp 3092-3103.

Liu, J., Chen, S., Zhou, Z.-H., & Tan, X. (2007). "Single Image Subspace for Face Recognition". *Proceedings of the 3rd international conference on Analysis and Modeling of Faces and Gestures (AMFG'07)* (pp. 205-219). Springer-Verlag.

Liu, J., Wang, W., & Yang, J. (2004). "A Framework for Ontology-Driven Subspace Clustering". *KDD'04* (pp. 623-628). Seattle: ACM.

Liu, T.-Y., Yang, Y., Wan, H., Zeng, H.-J., Chen, Z., & Ma, W.-Y. (2005). "Support Vector Machines Classification with A Very Large-scale Taxonomy". *SIGKDD Explorations, 7*(1), 36-43.

Lu, H., Plataniotis, K. N., & Venetsanopoulos, A. N. (2011). "A Survey of Multilinear Subspace Learning for Tensor Data". *Pattern Recognition, 44*, 1540-1551.

Manning, C., Raghavan, P., & Schutze, H. (2008). *“Introduction to Information Retrieval”.* Cambridge University Press.

Martin, B. (1995). “Instance-Based learning : Nearest Neighbor With Generalization,”. *Master Thesis, University of Waikato.* Hamilton, New Zealand.

McCallum, A., Rosenfeld, R., Mitchell, T., & Ng, A. Y. (1998). "Improving Text Classification by Shrinkage in a Hierarchy of Classes". *Proceedings of the Fifteenth International Conference on Machine Learning (ICML'98)* (pp. 359-367). Madison, Wisconsin, USA: Morgan Kaufmann Publishers Inc.

Moise, G., Sander, J., & Ester, M. (2006). "P3C: A Robust Clustering Algorithm". *Proceedings of the Sixth IEEE International Conference on Data Mining (ICDM'06)*, (pp. 414-425). Hong Kong.

Nayar, S. K., Nene, S. A., & Murase, H. (1996). "Subspace Methods for Robot Vision". *IEEE Transactions on Robotics and Automation, 12*(5), 750-758.

Panagakis, I., Benetos, E., & Kotropoulos, C. (2008). "Music Genre Classification: A Multilinear Approach". *The Ninth International Conference on Music Information Retrieval (ISMIR 2008)*, (pp. 583-588). Philadelphia.

Pang , B., & Lee, L. (2008). "Opinion Mining and Sentiment Analysis". *Foundations and Trends in Information Retrieval, 2*(1-2), 1-135.

Parsons, L., Haque, E., & Liu, H. (2004). “Subspace Clustering for High Dimensional Data : A Review”. *ACM SIGKDD Explorations Newsletter, Vol 6*(Issue 1), pp. 90 – 105.

Pernkopf, F. (2007). “Discriminative learning of Bayesian network classifiers”. *Proceedings of the 25th IASTED International Multi-Conference: artificial intelligence and applications*, (pp. 422-427).

Popescu, M., Balas, V., Perescu-Popescu, L., & Mastorakis, N. (2009). “MultiLayer Perceptron and Neural Networks”. *WSEAS Transactions on Circuits and Systems*, *Vol 8, Issue 7, July 2009*, pp. 579-588.

Qi, G.-J., Tian, Q., & Huang, T. (2011). "Locality-Sensitive Support Vector Machine by Exploring Local Correlation and Global Regularization". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2011)*, (pp. 841-848). Colorado Springs.

Qiu, X., Gao, W., & Huang, X. (2009). "Hierarchical Multi-Class Text Categorization with Global Margin Maximization". *Proceedings of ACL-IJCNLP 2009 Conference Short Papers* (pp. 165-168). Singapore: ACL.

Quinlan, J. (1993). *“C4.5 : Programs for Machine Learning”.* San Mateo, CA.: Morgan Kaufmann Publishers.

Rose, T., Stevenson, M., & Whitehead, M. (2002). “The Reuters Corpus Volume 1 - from Yesterday’s News to Tomorrow’s Language Resources”. *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC- 02),2002*, (pp. 827–833).

Ruggieri, S. (2002). “Efficient C4.5”. *IEEE Transactions on Knowledge and Data Engineering*, *Vol 14, Issue 2, March 2002*, pp. 438 – 444.

Ruiz, M., & Srinivasan, P. (1999). "Hierarchical Neural Networks for Text Categorization". *SIGIR 1999*.

Salakhutdinov, R., & Hinton, G. (2009). "Semantic Hashing". *International Journal of Approximate Reasoning, 50*(7), 969-978.

Schutze, H., & Silverstein, C. (1997). "Projections for Efficient Document Clustering". *SIGIR 1997* (pp. 74-81). Philadelphia: ACM.

Sebastiani, F. (2002, March). "Machine Learning in Automated Text Categorization". *ACM Computing Surveys, 34*(1), 1-47.

Sebastiani, F. (2005). "Text Categorization". In A. Zanasi (Ed.), *Text Mining and its Applications* (pp. 109-129). Southampton, U.K.: WIT Press.

Szepannek, G., & Luebke, K. (2004). "Different Subspace Classification". (C. Weighs, & W. Gaul, Eds.) *Studies in Classification, Data Analysis and Knowledge Organization: Classification - The Ubiquitous Challenge*, pp. 224-231.

Torkkola, K. (2001). "Linear Discriminant Analysis in Document Classification". *IEEE ICDM Workshop on Text Mining.*

Tulyakov, S., Jaeger, S., Govindaraju, V., & Doermann, D. (2008). "Review of Classifier Combination Methods". *Studies in Computatonal Intelligence (SCI), 90*, 361-386.

Verma, B. (1997). “Fast training of multilayer perceptrons”. *IEEE Transactions on Neural Networks*, *Vol 8, Issue 6, Nov 1997*, pp. 1314-1320.

Wang, H., Wang, C., Zhang, L., & Zhou, D. (2004). "Data Clustering Algorithm based on Binary Subspace Division". *Proceedings of the 2004 International Conference on Machine Learning and Cybernetics*, *2*, pp. 1249 - 1253.

Weigend, A. S., Weiner, E. D., & Pedersen, J. O. (1999, October). "Exploiting Hierarchy in Text Categorization". *Information Retrieval, 1*(3), 193-216.

Wermter, S. (1995). *“Hybrid Connectionist Natural Language Processing”.* Chapman and Hall, 1995.

Wermter, S., Panchev, C., & Arevian, G. (1999). “Hybrid Neural Plausibility Networks for News Agents”. *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, (pp. 93-98).

Wetzker, R., Umbrath, W., Hennig, L., Bauckhage, C., Alpcan, T., & Metze, F. (2008). Tailoring Taxonomies for Efficient Text Categorization and Expert Finding. *IEEE International Conference on Web Intelligence and Intelligent Agent Technology*, (pp. 459-462).

Xu, L., Krzyzak, A., & Suen, C. Y. (1992). "Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition". *IEEE Transactions on Systems, Man and Cybernetics, 22*(3).

Xu, Y., Shen, F., Ping, W., & Zhao, J. (2011). "TAKES: A Fast Method to Select Features in the Kernel Space". *20th ACM Conference on Information and Knowledge Management (CIKM ’11)* , (pp. 683-692). Glasgow.

Yan, J., Cheng, Q., Yang, Q., & Zhang, B. (2005). "An Incremental Subspace Learning Algorithm to Categorize Large Scale Text Data". *Proceedings of the 7th Asia- Pacific Web Conference (APWeb 2005), LNCS 3399*, (pp. 52-63). Shanghai.

Yan, J., Liu, N., Zhang, B., Yang, Q., Yan, S., & Chen, Z. (2006). "A Novel Scalable Algorithm for Supervised Subspace Learning". *The Sixth International Conference on Data Mining (ICDM’06)*, (pp. 721-730).

Yang, J., Yan, S., & Huang, T. (2009, February). "“Ubiquitously Supervised Subspace Learning". *IEEE Transactions on Image Processing, 18*(2), 241-249.

Yang, Y. (1999). "An Evaluation of Statistical Approaches to Text Categorization". *Journal of Information Retrieval, 1*, 67-88.

Yang, Y., & Liu, X. (1999). "A re-examination of text categorization methods". *Proceedings of the 22nd annual ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, (pp. 42-49).

Yang, Y., Zhang, J., & Kisiel, B. (2003). "A Scalability Analysis of Classifiers in Text Categorization". *Proceedings of the 26th annual international ACM SIGIR conference* *on Research and development in informaion retrieval (SIGIR '03)* (pp. 96-103). Toronto, Canada: ACM.

Yaslan, Y., & Cataltepe, Z. (2010). "Co-training with relevant random subspaces",. *Neurocomputing 73 (2010)*, pp 1652-1661.

Yiu, M. L., & Mamoulis, N. (2005, February). "Iterative Projected Clustering by Subspace Mining". *IEEE Transactions on Knowledge and Data Engineering, 17*(2), 176-189.

Zaki, M. J., Peters, M., Assent, I., & Seidl, T. (2007). "CLICKS: An effective algorithm for mining subspace clusters in categorical datasets". *Data & Knowledge Engineering, 60*, 51-70.

Zeimpekis, D., & Gallopoulos, E. (2005). “TMG : A MATLAB Toolbox for Generating Term Document Matrices from Text Collections”. In J. Kogan, & C. Nicholas (Eds.), *Book Chapter in Grouping Multidimensional Data: Recent Advances in Clustering.* Springer.

Zhou, L.-H., Liu, W.-Y., Xu, Y.-F., & Chen, H.-M. (2008). "Bayesian Network-Based Projected Clustering". *Proceedings of the Seventh International Conference on Machine Learning and Cybernetics*, (pp. 2651-2656). Kunming.

Appendix 1

List of Publications

The following is a list of publications related to this work:

Journal Publication:

1. Tripathi, N., Oakes, M. & Wermter, S. (2011). "Semantic subspace learning for text classification using hybrid intelligent techniques". *International Journal of Hybrid Intelligent Systems*, Vol. 8(2), pp. 99 -114, 2011.

Conference Publications:
(in reverse chronological order)

1. Tripathi, N., Oakes, M. & Wermter, S. (2012). "A Fast Subspace Text Categorization Method Using Parallel Classifiers". *13th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing 2012)*, Springer LNCS 7182, pp. 132-143, New Delhi, India, March 2012.
2. Tripathi, N., Oakes, M. & Wermter, S. (2011). "Hybrid Classifiers for Improved Subspace Learning of News Documents". *International Conference on Hybrid Intelligent Systems (HIS 2011)*, pp. 28-33, Malacca, Malaysia, Dec 2011.
3. Tripathi, N., Oakes, M. & Wermter, S. (2011). "Hybrid Parallel Classifiers for Semantic Subspace Learning". In Honkela, T., Duch, W., Girolami, M., Kaski, S., editors, Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN 2011), Part II, pp. 64-70, Espoo, Finland, June 2011.
4. Tripathi N., Wermter S., Hung C. & Oakes, M. (2010). "Semantic Subspace Learning with Conditional Significance Vectors". Proceedings of the International Joint Conference on Neural Networks (IJCNN/WCCI 2010), pp. 3670-3677, Barcelona, July 2010.

Appendix 2

Publications Attached

The following publications are attached here:

1. Tripathi N., Wermter S., Hung C. & Oakes, M. (2010). “Semantic Subspace Learning with Conditional Significance Vectors”. Proceedings of the International Joint Conference on Neural Networks (IJCNN/WCCI 2010), pp. 3670-3677, Barcelona, July 2010.

2. Tripathi, N., Oakes, M. & Wermter, S. (2011). “Semantic subspace learning for text classification using hybrid intelligent techniques”. *International Journal of Hybrid Intelligent Systems*, Vol. 8(2), pp. 99 -114, 2011.

Semantic Subspace Learning with Conditional Significance Vectors

Nandita Tripathi, Stefan Wermter, Chihli Hung, and Michael Oakes

Abstract — **Subspace detection and processing is receiving more attention nowadays as a method to speed up search and reduce processing overload. Subspace Learning algorithms try to detect low dimensional subspaces in the data which minimize the intra-class separation while maximizing the inter-class separation. In this paper we present a novel technique using the maximum significance value to detect a semantic subspace. We further modify the document vector using conditional significance to represent the subspace. This enhances the distinction between classes within the subspace. We compare our method against TFIDF with PCA and show that it consistently outperforms the baseline with a large margin when tested with a wide variety of learning algorithms. Our results show that the combination of subspace detection and conditional significance vectors improves subspace learning.**

I. INTRODUCTION

Many learning algorithms do not perform well with high-dimensional data due to the *curse of dimensionality* [1]. Additional dimensions spread out the points making distance measures less useful. In very high dimensions, objects in a dataset are nearly equidistant from each other. Therefore, methods are needed that can discover clusters in various subspaces of high dimensional datasets [2].

Subspace learning methods are therefore nowadays being increasingly researched and applied to web document classification, image recognition and data clustering. Among subspace learning methods, Principal Component Analysis (PCA) [3] and Linear Discriminant Analysis (LDA) [4] are well known traditional methods. LDA is a supervised method whereas PCA is unsupervised. Other methods include ISOMAP, Locally Linear Embedding (LLE), Neighborhood Preserving Embedding (NPE), Laplacian Eigen Maps, Nonparametric Discriminant Analysis, Marginal Fisher Analysis and Local Discriminant Embedding [5]. Furthermore, the Supervised Kampong Measure (SKM) [6] is an incremental subspace learning method.

The objective of all these algorithms is to minimize the intra-class distance while maximizing the inter-class separation. However, as the number of feature dimensions increases, the computational complexity for these algorithms increases dramatically. For instance, the computational complexity of PCA is $O(p^2n)+O(p^3)$ where p is number of data dimensions and n is the number of data points [7]. In other approaches, Wang et al [8] use an RD-Quadtree to subdivide the data space and show that their RD–Quadtree-based clustering algorithm has better results for high-dimensional data than the well-known K–means algorithm. Finally, Hinton & Salakhutdinov [9] have proposed the concept of Semantic Hashing where documents are mapped to memory addresses in such a way that semantically similar documents are located at nearby addresses. The majority of these methods have high computational complexity and as such cannot quickly focus the search when the amount of data is very large.

We present here a novel method of subspace detection and show that it improves learning without lengthy computations. We use the semantic significance vector [10], [11] to incorporate semantic information in the document vectors. We compare the performance of these vectors against that of TFIDF vectors. The dimensionality of TFIDF vectors is reduced using PCA to produce a vector length equal to that of the semantic significance vectors. Our experiments were performed on the Reuters corpus (RCV1) using the first two levels of the topic hierarchy. Our method achieves the objective of the other subspace learning algorithms i.e. decreasing intra-class distance while increasing inter-class separation but without the associated computational cost. Subspace detection is done in O(k) time where k is the number of level 1 topics and thus can be very effective where time is critical for returning search results.

II. METHODOLOGY OVERVIEW AND OVERALL ARCHITECTURE

The topic codes in the Reuters Corpus [12] represent the subject areas of each news story. They are organized into four hierarchical groups, with four top-level nodes: Corporate/Industrial (CCAT), Economics (ECAT), Government/Social (GCAT) and Markets (MCAT). Under each top-level node there is a hierarchy of codes,

Final Manuscript received April 26,2010.

Nandita Tripathi is with the Department of Computing, Engineering and Technology, University of Sunderland, St Peters Way, Sunderland SR6 0DD, United Kingdom (e-mail: bf39rv@ student.sunderland.ac.uk)

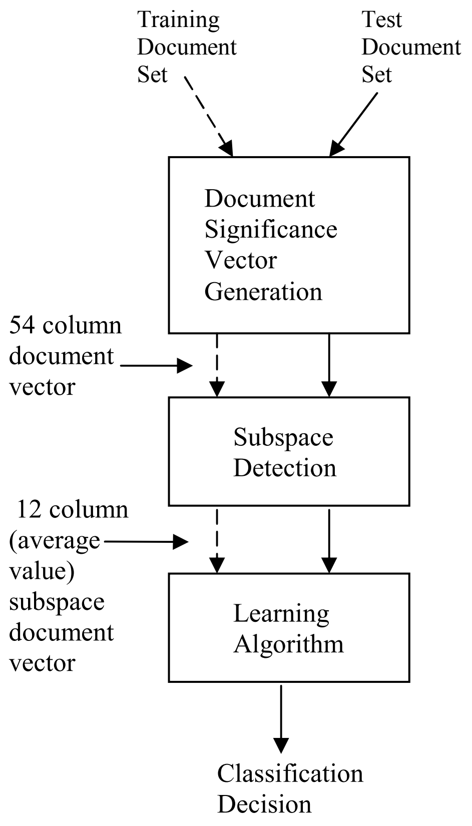
Stefan Wermter is with Institute for Knowledge Technology, Department of Computer Science, University of Hamburg, Vogt Koelln, Str. 30, 22527 Hamburg , Germany (e-mail: wermter@informatik.uni-hamburg.de)

Chihli Hung is with the Department of Information Management, Chung Yuan Christian University, Chung-Li, Taiwan 32023 R.O.C. (email: chihli@cycu.edu.tw)

Michael Oakes is with the Department of Computing, Engineering and Technology, University of Sunderland, St Peters Way, Sunderland SR6 0DD, United Kingdom (e-mail: Michael.Oakes@sunderland.ac.uk).

with the depth of each represented by the length of the code.

Ten thousand headlines along with their topic codes were extracted from the Reuters Corpus. These headlines were chosen so that there was no overlap at the first level categorization. Each headline belonged to only one level 1 category. At the second level, since most headlines had multiple level 2 subtopic categorizations, the first subtopic was taken as the assigned subtopic. Thus each headline had two labels associated with it – the main topic (Level 1) label and the subtopic (Level 2) label. Headlines were then pre-processed to separate hyphenated words. Dictionaries with term frequencies were generated based on the TMG toolbox [13]. These were then used to generate the Semantic Significance Vector representation [10], [11] for each document. Two different variations of vector representations were used – the Full Significance Vector representation and the new Conditional Significance Vector representation. Masking of the vector elements was done by setting them to zero value. Different levels of masking were examined to generate a total of five different datasets. Each dataset



Training Document Set
Test Document Set
Document Significance Vector Generation
54 column document vector
Subspace Detection
12 column (average value) subspace document vector
Learning Algorithm
Classification Decision

Fig 1. Semantic Subspace Learning Architecture

was then randomised and divided into two equal sets for training and testing, each comprising of 5000 document vectors. Fig 1 shows the semantic subspace learning architecture.

The WEKA machine learning workbench [14] was used to examine this architecture and representations with various learning algorithms. Ten algorithms were compared for our representations to examine the different categories of classification algorithms. Classification Accuracy, which is a comparison of the predicted class to the actual class, was recorded for each experiment run.

III. Steps for Data Processing and Data Generation for Experiments

3.1 Text Data Preprocessing

10,000 Reuters headlines were used in these experiments. The Level 1 categorization of the Reuters Corpus divides the data into four main topics. These main topics and their distribution in the data along with the number of subtopics of each main topic in this data set are given in Table 1.

Level 2 categorization further divides these into subtopics. Here we took the direct (first level nesting) subtopics of each main topic occurring in the 10,000 headlines. A total of 50 subtopics were included in these experiments. Some of these subtopics with their numbers present are shown in Table 2. Since all the headlines had multiple subtopic assignment e.g. C11/C15/C18, only the first subtopic e.g. C11 was taken as the assigned subtopic.

Table 1: Reuters Level 1 Topics

No	Main Topic	Description	Number Present	No. of Sub-topics
1	CCAT	Corporate/ Industrial	4600	18
2	ECAT	Economics	900	8
3	GCAT	Government/ Social	1900	20
4	MCAT	Markets	2600	4

Table 2: Some Reuters Level 2 subtopics used for our experiments.

Main Topic	Sub Topic	Description	Number Present
CCAT	C17	Funding/ Capital	377
CCAT	C32	Advertising/ Promotion	10
CCAT	C41	Management	130
ECAT	E12	Monetary/ Economic	107
ECAT	E21	Government Finance	377
ECAT	E71	Leading Indicators	87
GCAT	G15	European Community	38
GCAT	GPOL	Domestic Politics	197
GCAT	GDIP	International relations	215
GCAT	GENV	Environment	30
MCAT	M11	Equity Markets	617
MCAT	M14	Commodity Markets	1050

3.2 Semantic Significance Vector Generation

We use a vector representation which looks at the significance of the data and weighs different words according to their significance for different topics. References [10] and [11] have introduced the concept of semantic significance vectors. Significance Vectors are determined based on the frequency of a word in different semantic categories. A modification of the significance vector called the semantic vector uses normalized frequencies. Each word *w* is represented with a vector $(c_1,c_2,..,c_n)$ where c_i represents a certain semantic category and *n* is the total number of categories. A value $v(w, c_i)$ is calculated for each element of the semantic vector as the normalized frequency of occurrences of word *w* in semantic category c_i (the normalized category frequency), divided by the normalized frequency of occurrences of the word *w* in the corpus (the normalized corpus frequency):

$$v(w,c_i) = \frac{Normalised\ Frequency\ of\ w\ in\ c_i}{\sum_k Normalised\ Frequency\ of\ w\ in\ c_k}$$

$$where$$

$$k \in \{1..n\}$$

For each document, the document semantic vector is obtained by summing the semantic vectors for each word in the document and dividing by the total number of words in the document. This is the version of the semantic significance vector used in our experiments. Henceforth it is simply referred to as *Significance Vector*. The TMG Toolbox [13] was used to generate the term frequencies for each word in each headline. The word vector consisted of 54 columns for 4 main topics and 50 subtopics. While calculating the significance vector entries for each word, its occurrence in all subtopics of all main topics was taken into account - hence called *Full Significance Vector*.

We also generated vectors to observe whether results obtained with Full Significance can be improved by modifying the significance vectors to reflect the subspace which is being processed. Here again the word vector consisted of 54 columns for 4 main topics and 50 subtopics. However, while calculating the significance vector entries for each word, its occurrence in all subtopics *of only a particular main topic* was taken into account - henceforth called *Conditional Significance Vector*.

3.3 Data Sets Generation

As will be described below, datasets for five different vector representations were generated. The Full Significance Vectors were processed in different ways to generate four different data sets. The fifth set was the Conditional Significance Vector dataset.

3.3.1 No Mask Full Significance Data Set

For each vector the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were ignored leaving a vector with 50 columns representing 50 subtopics. The order of the data vectors was then randomised and divided into two sets – training set and testing set of 5000 vectors each.

3.3.2 Mask 1 Full Significance Data Set

For each vector the numerical entries in the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were compared. The topic with the minimum numerical value entry was identified. Then the entries for all subtopics belonging to this main topic were masked i.e. set to zero. Finally, the first four columns representing four main categories were deleted. The resultant vector had 50 columns representing 50 subtopics but the subtopic entries for the topic with least significance value had been masked to zero. The average number of relevant columns was then 38. The dataset was then randomised and divided into two sets – training set and testing set of 5000 vectors each.

3.3.3 Mask 2 Full Significance Data Set

As above, the numerical entries in the first four columns of each vector representing four main topics CCAT, ECAT, GCAT and MCAT were compared. The main topics with the two smallest numerical value entries were identified. Then the entries for all subtopics belonging to these two main topics were masked i.e. set to zero. Then, the first four columns representing four main categories were ignored. The resultant vector had 50 columns representing 50 subtopics but the subtopic entries for the two topics with the two smallest significance values had been masked to zero. The average number of relevant columns in this case became 25. The masked dataset was then randomised and divided into training and testing sets of 5000 vectors each.

3.3.4 Mask 3 Full Significance Data Set

Here again, the numerical entries in the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were compared. The topics with the three smallest numerical value entries were identified. Then the entries for all subtopics belonging to these three topics were masked i.e. set to zero. Finally, the first four columns representing four main categories were deleted. The resultant vector had 50 columns representing 50 subtopics but the subtopic entries for the three main topics with least significance value, 2nd least significance value and 3rd least significance value had been masked to zero. Since there are four main topics in the Reuters corpus, this has the same effect as allowing only the subtopics of the main topic with the maximum significance value in the resultant vector while masking out all the rest. The average number of relevant columns here was 12. Again the dataset was randomised and divided into training set and testing set of 5000 vectors each.

3.3.5 Mask 3 Conditional Significance Data Set

In this case, while calculating the significance vector entries for each word in a subtopic, its occurrence in all subtopics *of only a particular main topic* was taken into account - hence called *conditional significance vector*. Here, when calculating significance values for C11, C12, etc, the topics considered were only the subtopics of CCAT. Similarly for M11, M12, etc only MCAT subtopics were considered. For each word, four separate conditional significance sub-vectors were generated for the four main Reuters topics. These sub-vectors were then concatenated together along with the significance value entries for the four main topics to form the 54 column word vector. The Conditional Significance document vector was generated by summing the conditional significance word vectors for each word appearing in the document and then dividing by the total number of words in the document. This vector representation is used to measure the significance of a word within a particular main topic. Hence only the subtopic entries for the main topic with maximum value entry were allowed. All the subtopic entries belonging to the other 3 main topics were masked out. The dataset was then randomised and divided into two sets – training set and testing set of 5000 vectors each.

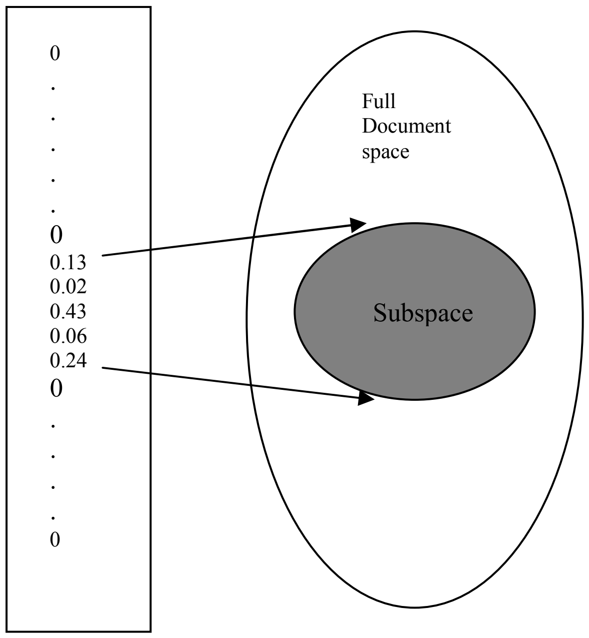


Fig 2: Mapping of Conditional Significance Vector to relevant subspace.

Fig 2 shows the conceptual diagram for the conditional significance vector while Fig 3 shows the Conditional Significance Vector (CSV) for two different Reuters headlines. The Mask 3 Full Significance Vector (FSV) and Conditional Significance Vector (CSV) values for each of these two headlines are given below for comparison. The main topic label and subtopic label are shown at the end of each vector. As can be seen, the vector entries are boosted in the case of CSV – thus helping to differentiate between subtopics within the subspace

Headline 1
0......0,0.20,0.03,0.04,0.02, MCAT/M11 : FSV
0......0,0.59,0.11,0.20,0.08, MCAT/M11 : CSV

Headline 2
0….0,0.03,0.05,0.04,0.0099,0.01,0.0073, 0.25,0.0069, 0.....0,ECAT/E51 : FSV
0….0,0.13,0.13,0.10,0.0100,0.02,0.0300,0.52,0.0300, 0….0,ECAT/E51 : CSV

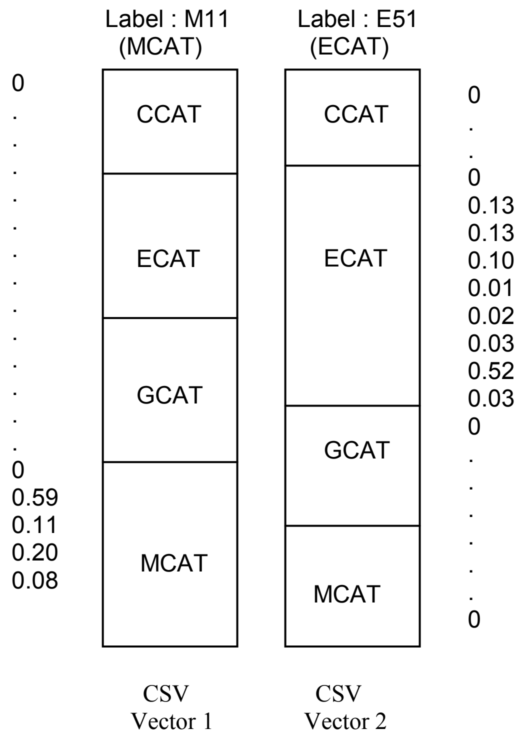


Fig 3: Conditional Significance Vectors showing non-zero entries for relevant subspace

3.4 TFIDF Vector generation

The TMG toolbox [13] was used to generate TFIDF vectors for the ten thousand Reuters headlines used in these experiments. Dimensionality reduction was done using PCA with the same toolbox. The number of dimensions was chosen as 50 for PCA to have vectors similar in size to the significance vectors generated earlier. The dataset was then randomized and divided into two sets - training and test of 5000 vectors each.

3.5 Classification Algorithms

Ten Classification algorithms were tested with our datasets namely Random Forest, C4.5, Bagging, LogitBoost, Classification via Regression, Multilayer Perceptron, BayesNet, IBk, NNge and PART. Random forests [15] are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently. C4.5 [16] is an inductive tree algorithm with two pruning methods : subtree replacement and subtree raising. Bagging [17] is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. LogitBoost [18] performs classification using a regression scheme as the base learner, and can handle multi-class problems. In Classification via Regression [19], one regression model is built for each class value. Multilayer Perceptron [20] is a neural network which uses backpropagation for training. BayesNet [21] implements Bayes Network learning using various search algorithms and quality measures. IBk [22] is a k-nearest neighbour classifier which can select an appropriate value of k based on cross-validation and can also do distance weighting. NNge [23] is a nearest neighbor like algorithm using non-nested generalized exemplars (which are hyperrectangles that can be viewed as if-then rules). A PART [24] decision list uses separate-and-conquer. It builds a partial C4.5 decision tree in each iteration and makes the best leaf into a rule. Table 3 shows the different classification algorithms used with their default parameters in Weka.

Table 3: Classification Algorithms and their default parameters in Weka

No.	Algorithm	Parameters
1.	Random Forest	NumTrees = 10
2.	J48 (C4.5)	Confidence factor=0.25, MinNumObj=2, NumFolds=3, Subtree raising =true
3.	Bagging	BagSizePerc=100, NumIterations=10, BaseClasifier=REP Tree
4.	Classification via Regression	Classifier=M5P
5.	LogitBoost	NumIterations =10, NumRuns =1, Shrinkage =1.0, Weight threshold =100, BaseClassifier = DecisionStump
6.	PART	Confidence factor=0.25, MinNumObj=2, NumFolds=3
7.	IBk	Knn=1, No cross validation, No distance weighting
8.	BayesNet	Estimator=SimpleEstimator, Search algorithm=K2
9.	NNge	NumAttemptsGeneOption=5 NumFoldersMIOption=5
10.	Multilayer Perceptron	LearningRate=0.3, Momentum=0.2, Training time=500, RandomValidation threshold=20

All these algorithms can cope with different sized categories. This takes care of the different number of instances present for each category in Table 1.

IV. RESULTS AND ANALYSIS

Two sets of experiments were run to test learning at the first two levels of Reuters topic categorization. Ten runs of each algorithm with different seed values (wherever applicable) were taken for each vector representation. Four algorithms (Classification via Regression, IBk, BayesNet and NNge) did not have the option for entering a random seed value in Weka. Three algorithms (C4.5, LogitBoost and PART) had an option for entering random seed value but the results for all 10 runs were identical. Only three algorithms (Random Forest, Bagging and Multilayer Perceptron) showed variance in the classification accuracy values. The average and variance of the classification accuracy for 10 runs with different seed values was calculated for each algorithm. The abbreviations for the various options are given below:

FS_0: Full Significance with No Mask
FS_1: Full Significance with Mask 1
FS_2: Full Significance with Mask 2
FS_3: Full Significance with Mask 3
CS_3: Conditional Significance with Mask 3
TFIDF/PCA: TFIDF with PCA reduction

The Algorithm Index is as follows:

1. Random Forest
2. J48 (C4.5)
3. Bagging
4. Classification via Regression
5. LogitBoost
6. PART
7. IBk
8. BayesNet
9. NNge
10. Multilayer Perceptron

4.1 Level 1 Testing

The Full Significance Vector with four variations – No Mask, Mask 1, Mask 2 and Mask 3 and the Conditional Significance Vector with Mask 3 were used with only the main topic labels i.e. CCAT, ECAT, GCAT and MCAT. The TFIDF/PCA vectors with main topic labels were used for comparison. The algorithms given above were run using 5000 training vectors and 5000 test vectors for each case. Table 4a shows the average accuracy values while Table 4b shows the variance in accuracy values for the test cases.

Table 4a: Main Topic Average Classification Accuracy (%) for test vectors
Bold Font (big) – best performance; **Bold Font** (small) - 2nd best performance

*No.	FS_0	FS_1	FS_2	FS_3	CS_3	TFIDF/PCA
1.	91.17	90.67	91.45	**96.45**	**96.45**	79.46
2.	92.46	91.02	92.40	**95.72**	**96.10**	73.58
3.	92.24	91.95	93.54	**96.39**	**96.29**	78.89
4.	92.10	94.94	94.72	**96.28**	**96.78**	77.54
5.	92.30	92.22	90.96	**96.24**	**96.38**	72.20
6.	93.46	92.86	92.20	**95.92**	**95.60**	74.14
7.	**96.84**	**96.74**	95.28	95.44	95.94	76.74
8.	83.58	81.26	71.70	**96.26**	**96.30**	59.62
9.	95.66	95.58	89.92	**96.64**	**96.34**	73.72
10.	**96.54**	96.40	95.31	96.49	**97.43**	79.77

*Algorithm No

Table 4b: Main Topic Classification Accuracy Variance for test vectors
Bold Font (big) – best performance

*No.	FS_0	FS_1	FS_2	FS_3	CS_3	TFIDF/PCA
1.	0.227	0.236	0.123	0.018	**0.011**	0.120
2.	0	0	0	0	0	0
3.	0.234	0.084	0.042	**0.003**	**0.003**	0.224
4.	0	0	0	0	0	0
5.	0	0	0	0	0	0
6.	0	0	0	0	0	0
7.	0	0	0	0	0	0
8.	0	0	0	0	0	0
9.	0	0	0	0	0	0
10.	0.109	0.115	0.095	0.062	**0.042**	0.742

*Algorithm No

In Table 4a, all algorithms except IBk (No. 7) show that the maximum masking option (Mask 3) gives the best result. This indicates that the maximum significance value is a good indicator of the relevant subspace. The best results are divided between FS_3 and CS_3 for different algorithms. Table 4b shows that the minimum variance is also given by the maximum masking option (Mask 3). The best result is given by CS_3.

Table 5a: Subtopic Average Classification Accuracy (%) for test vectors
Bold Font (big) – best performance; **Bold Font** (small) - 2nd best performance

*No.	FS_0	FS_1	FS_2	FS_3	CS_3	TFIDF/ PCA
1.	82.11	80.69	74.69	**88.55**	**90.60**	57.37
2.	87.90	87.62	78.50	**88.90**	**90.42**	49.16
3.	86.68	87.04	79.51	**89.53**	**92.06**	57.51
4.	**92.12**	91.94	83.32	91.36	**92.98**	56.02
5.	**92.32**	92.10	83.88	91.16	**92.62**	52.98
6.	87.18	87.98	77.20	**88.78**	**90.24**	50.44
7.	**90.84**	90.58	81.76	89.66	**91.22**	55.52
8.	68.52	61.98	52.18	**86.84**	**89.04**	46.74
9.	**91.30**	91.16	82.34	90.96	**92.42**	54.82
10.	**91.96**	91.86	82.07	91.39	**92.39**	58.84

*Algorithm No

Table 5b: Subtopic Classification Accuracy Variance for test vectors
Bold Font (big) – best performance

*No.	FS_0	FS_1	FS_2	FS_3	CS_3	TFIDF/ PCA
1.	0.545	1.239	0.264	**0.101**	0.146	0.202
2.	0	0	0	0	0	0
3.	0.233	0.241	0.151	0.030	**0.028**	0.147
4.	0	0	0	0	0	0
5.	0	0	0	0	0	0
6.	0	0	0	0	0	0
7.	0	0	0	0	0	0
8.	0	0	0	0	0	0
9.	0	0	0	0	0	0
10.	0.141	0.163	0.746	**0.026**	0.059	0.320

*Algorithm No

4.2 Level 2 Testing

Here, the Full Significance Vector with four variations – No Mask, Mask 1, Mask 2 and Mask 3 and the Conditional Significance Vector with Mask 3 were used with the subtopic labels. The TFIDF/PCA vectors with subtopic labels were used for comparison here. The same algorithms as given above were run using 5000 training vectors and 5000 test vectors for each case. The results shown in Table 5a are the average accuracy values for the test cases. This subtopic accuracy table shows the accuracy values obtained by applying classification algorithms after subspace branching. So this is a combined performance of level 1 and level 2. The Conditional Significance Vector representation with maximum masking option (Mask 3) gives the best average accuracy result with all algorithms. Table 5b shows that the minimum variance is again given by the maximum masking option (Mask 3). The best variance values are split between FS_3 and CS_3 here.

The Mask 3 option consistently shows the best results at level 1 and level 2. This shows that the maximum significance value is successful in identifying the relevant subspace (level 1 topic). Since the Conditional Significance Vector with Mask 3 option encodes the subspace within the vector itself, the subtopic accuracy table shows the combined effect of branching at level one and applying the classification algorithms at level 2. Consistent maximum accuracy obtained at level 2 by the conditional significance vector with all the algorithms shows that conditional significance is successful in differentiating between subtopics within a data subspace. Thus our vector representation is unique in that it incorporates both subspace branching and subspace learning in the same step.

V. CONCLUSION

This work is an effort to explore semantic subspace learning with the overall objective of improving document retrieval in a vast document space. Our experiments on the Reuters Corpus show that the maximum significance value has potential in identifying the main (Level 1) topic of a document. They also show that modifying the significance vector (conditional significance) to process only the subspace improves learning within the subspace. Thus the combination of branching on maximum significance value along with using conditional significance improves subspace learning. The subspace detection is done by processing a single document vector. This method is independent of the total number of data samples and only compares the level 1 topic entries. The time complexity is thus O(k) where k is the number of level 1 topics.

The novelty of our approach is in the vector representation. In the document conditional significance vector generated by the subspace detection step, the subspace is encoded in the vector (the non-zero values represent the subspace). Secondly, the numerical significance values in the word conditional significance vector denote the significance of a particular word for different subtopics *within* that subspace. Since the document vector is a summation of the word vectors, this helps in differentiating between topics within a given subspace (between subtopics of a main topic in case of Reuters Corpus) thus enhancing subspace learning. In this work, the word significance vectors were calculated using only term frequencies. For further work, the effect of a global weighting measure like Inverse Document Frequency (IDF) on the word weights can be explored.

REFERENCES

[1] J.H. Friedman,"On Bias, Variance, 0/1—Loss, and the Curse-of-Dimensionality," In Data Mining and Knowledge Discovery**,** Volume 1, Issue 1, 1997, pp 55 - 77

[2] L. Parsons, E. Haque & H. Liu, "Subspace Clustering for High Dimensional Data : A Review," In ACM SIGKDD Explorations Newsletter**,** Vol 6, Issue 1, 2004, pp 90 – 105

[3] I. Joliffe , "Principal Component Analysis," New York: Springer-Verlag, 1986

[4] K. Fukunaga, "Introduction to statistical pattern recognition," 2nd edition , Academic Press, New York, 1990.

[5] J.Yang, S.Yan & T.Huang, "Ubiquitously Supervised Subspace Learning," In IEEE Transactions on Image Processing, Vol 18, No 2, Feb 2009, pp 241 – 249.

[6] J.Yan, N. Liu, B. Zhang, Q.Yang, S.Yan & Z.Chen, "A Novel Scalable Algorithm for Supervised Subspace Learning," In Proceedings of the Sixth International Conference on Data Mining (ICDM'06), 2006, pp 721- 730

[7] D. Fradkin & D. Madigan, "Experiments with Random Projections for Machine Learning," In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003, pp 517-522

[8] H. Wang, C. Wang, L. Zhang & D. Zhou, "Data Clustering Algorithm based on Binary Subspace Division, " In Proceedings of 2004 International Conference on Machine Learning and Cybernetics, Volume 2, Issue, 26- 29 Aug. 2004, pp 1249 - 1253

[9] G.Hinton & R. Salakhutdinov , "Semantic Hashing, " In International Journal of Approximate Reasoning, Volume 50, Issue 7, July 2009, pp 969-978

[10] S.Wermter, C. Panchev & G. Arevian, "Hybrid Neural Plausibility Networks for News Agents," In Proceedings of the Sixteenth National Conference on Artificial Intelligence, 1999, pp 93-98

[11] S.Wermter , "Hybrid Connectionist Natural Language Processing," Chapman and Hall. 1995

[12] T. Rose, M. Stevenson, and M. Whitehead, "The Reuters Corpus Volume 1 - from Yesterday's News to Tomorrow's Language Resources," In Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-02),2002, pp 827–833.

[13] D. Zeimpekis and E. Gallopoulos, "TMG : A MATLAB Toolbox for Generating Term Document Matrices from Text Collections, " Book Chapter in Grouping Multidimensional Data: Recent Advances in Clustering, J. Kogan and C. Nicholas, eds., Springer, 2005

[14] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. Witten, "The WEKA Data Mining Software: An Update, " In ACM SIGKDD Explorations Newsletter, Volume 11, Issue 1, July 2009, pp 10-18.

[15] L. Breiman, "Random Forests," In Machine Learning 45(1), Oct. 2001, pp 5-32

[16] J.R. Quinlan, "C4.5 : Programs for Machine Learning," Morgan Kaufmann Publishers, San Mateo, CA. 1993

[17] L. Breiman, "Bagging predictors," In Machine Learning, 24(2), 1996, pp 123-140.

[18] J. Friedman , T. Hastie and R. Tibshirani , "Additive Logistic Regression: a Statistical View of Boosting, " Technical report. Stanford University. 1998

[19] E. Frank, Y. Wang, S. Inglis, G. Holmes, and I. H. Witten, "Using model trees for classification," In Machine Learning, Vol 32, No.1, 1998, pp. 63-76.

[20] B.Verma, "Fast training of multilayer perceptrons, " In IEEE Transactions on Neural Networks, Vol 8, Issue 6, Nov 1997 pp 1314-1320.

[21] F. Pernkopf, "Discriminative learning of Bayesian network classifiers, " In Proceedings of the 25th IASTED International Multi-Conference: artificial intelligence and applications, 2007, pp 422-427

[22] D. Aha and D. Kibler , " Instance-based learning algorithms," In Machine Learning, vol.6, 1991, pp. 37-66.

[23] B. Martin, "Instance-Based learning : Nearest Neighbor With Generalization", Master Thesis, University of Waikato, Hamilton,
New Zealand, 1995

[24] E. Frank and I. H. Witten, "Generating Accurate Rule Sets Without Global Optimization," In Shavlik, J., ed., Machine Learning: Proceedings of the Fifteenth International Conference, Morgan Kaufmann Publishers, 1998

International Journal of Hybrid Intelligent Systems 8 (2011) 99–114
DOI 10.3233/HIS-2011-0137
IOS Press

Semantic subspace learning for text classification using hybrid intelligent techniques

Nandita Tripathi[a,*], Michael Oakes[a] and Stefan Wermter[b]
[a]*University of Sunderland, UK*
[b]*University of Hamburg, Germany*

Abstract. A vast data repository such as the web contains many broad domains of data which are quite distinct from each other e.g. medicine, education, sports and politics. Each of these domains constitutes a subspace of the data within which the documents are similar to each other but quite distinct from the documents in another subspace. The data within these domains is frequently further divided into many subcategories. In this paper we present a novel hybrid parallel architecture using different types of classifiers trained on different subspaces to improve text classification within these subspaces. The classifier to be used on a particular input and the relevant feature subset to be extracted is determined dynamically by using maximum significance values. We use the conditional significance vector representation which enhances the distinction between classes within the subspace. We further compare the performance of our hybrid architecture with that of a single classifier – full data space learning system and show that it outperforms the single classifier system by a large margin when tested with a variety of hybrid combinations on two different corpora. Our results show that subspace classification accuracy is boosted and learning time reduced significantly with this new hybrid architecture.

Keywords: Parallel classifiers, hybrid classifiers, subspace learning, significance vectors, maximum significance

1. Introduction

The web is an almost infinite data repository. It contains a large number of data domains which are quite distinct from each other. A few examples of these are medicine, education, sports and politics. The data within these domains is frequently further subdivided into many levels of categories. These domains constitute different subspaces of data which can be processed as independent entities.

The *curse of dimensionality* [11] degrades the performance of many learning algorithms. Very high dimensions reduce the effectiveness of distance measures and blur the cluster boundaries within subspaces. Therefore, we need ways to discover clusters in different subspaces of datasets which are represented with a high number of dimensions [19].

Subspace analysis lends itself naturally to the idea of hybrid classifiers. Since each subspace can be viewed as an independent dataset, different classifiers can be used to process different subspaces. Each subspace can be processed by a classifier best suited to the characteristics of that particular subspace. Instead of using the complete set of full space feature dimensions, classifier performances can be boosted by using only a subset of the dimensions. The method of choosing an appropriate reduced set of dimensions is an active research area [14].

The use of Random Projections in dimensionality reduction has also been explored. Random Projections and PCA were compared on different datasets and ma-

[*]Corresponding author: Nandita Tripathi, c/o Dr. Michael Oakes, David Goldman Informatics Centre, School of Computing and Technology, University of Sunderland, Sunderland SR6 0DD, United Kingdom. Tel.: +44 191 515 3631; Fax: +44 191 515 2781; E-mail: Nandita.Tripathi@hotmail.com.

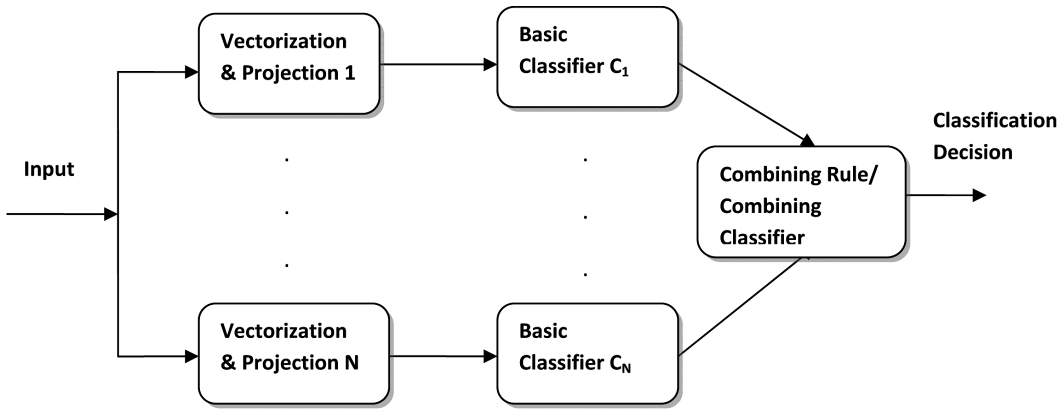

Input
Vectorization & Projection 1
Vectorization & Projection N
Basic Classifier C_1
Basic Classifier C_N
Combining Rule/ Combining Classifier
Classification Decision

Fig. 1. A combined classifier.

chine learning algorithms by Fradkin and Madigan [6]. They concluded that the performance of PCA was consistently better than that of Random Projections (RP) but RP was more efficient computationally and it was best suited with nearest neighbor methods. In the Random Subspace Method (RSM) [32], classifiers were trained on randomly chosen subspaces of the original input space and the outputs of the models were then combined. However random selection of features does not guarantee that the selected inputs have necessary distinguishing information. Several variations of RSM have been proposed by various researchers such as Relevant random feature subspaces for co-training (Rel-RASCO) [34], Not-so-Random Subspace Method (NsRSM) [23] and Local Random Subspace Method [28].

The performance of different types of classifiers (Bayesian, Tree based, Neural Networks, etc.) can be improved by combining them with various types of combining rules. In one method of classifier combination, several classifiers of different types operate on the same data and produce their individual classification outputs. A combination rule or combining classifier is then applied to the outputs of these participating classifiers to produce the final classification decision. In another method of classifier combination, many classifiers of the same or different types operate on different portions of the input data space. The combining classifier decides which part of the input data has to be applied to which base classifier. Two special types of classifier combinations are Bagging [15] and Boosting [25] which use a large number of primitive classifiers of the same type (e.g. a decision stump) on weighted versions of the original data. Figure 1 shows a general combined classifier.

Many experiments were conducted on combining classifiers by Duin and Tax [26] and it was reported that best performance is achieved by combining both, different feature sets and different classifiers. Several researchers have studied classifier combinations with respect to text categorization. In one method [13], text and metadata were considered as separate descriptions of the same object. These descriptions were classified by their independent classifiers and the classification outputs combined to give a final classification decision. Another text categorization method [20] was based on a hierarchical array of neural networks. In this case, the expert networks are specialized in recognizing documents corresponding to a specific category. The problem of large class imbalances in text classification tasks was addressed by using a mixture of experts framework [1]. Here different experts are trained on datasets sampled at different rates. Both oversampling and under sampling is used in this case.

In the real world, documents can be divided into major semantic subspaces with each subspace having its own unique characteristics. The above research does not take into account this division of documents into different semantic subspaces. Therefore we present here a novel hybrid parallel architecture (Fig. 2) which takes advantage of the different semantic subspaces existing in the data. We further show that this new hybrid parallel architecture improves subspace classification accuracy as well as significantly reduces training time. For this architecture, we test various hybrid combinations of classifiers using the conditional significance vector representation [24] which is a variation of the semantic significance vector [30,31] to incorporate semantic information in the document vectors. The conditional significance vector enhances the distinction between subtopics within a given main topic. The re-

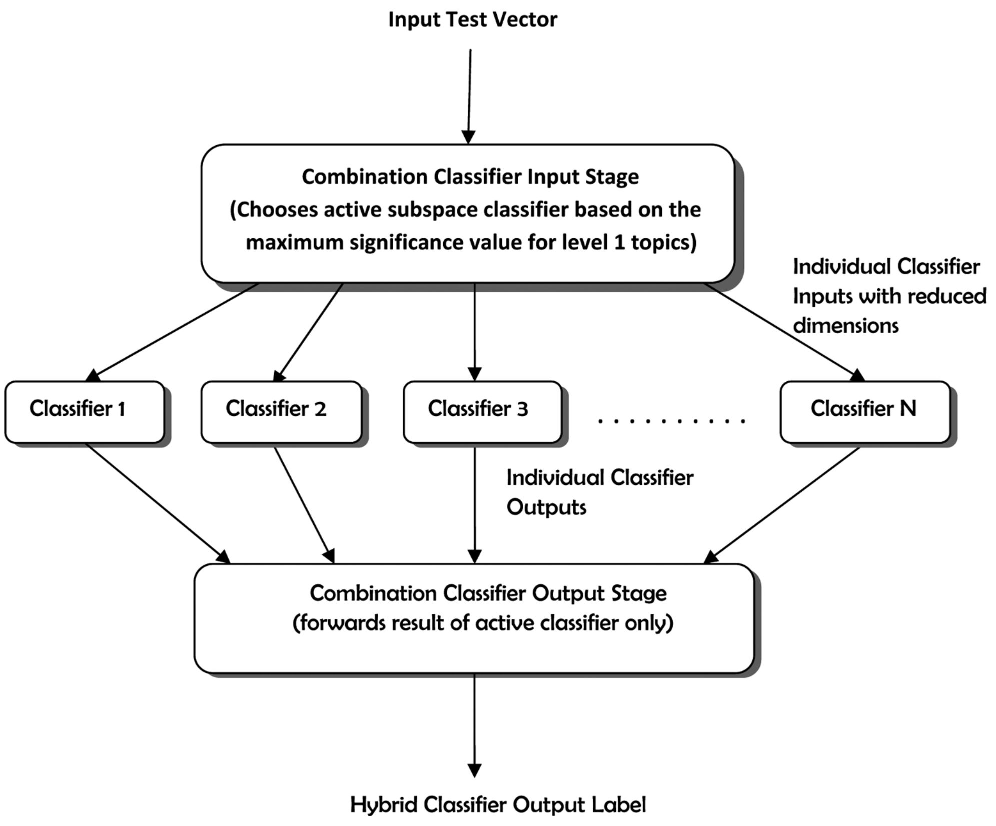

Input Test Vector
Combination Classifier Input Stage
(Chooses active subspace classifier based on the maximum significance value for level 1 topics)
Individual Classifier Inputs with reduced dimensions
Classifier 1
Classifier 2
Classifier 3
. . . .
Classifier N
Individual Classifier Outputs
Combination Classifier Output Stage
(forwards result of active classifier only)
Hybrid Classifier Output Label

Fig. 2. Hybrid parallel classifier architecture for subspace learning.

gion of the test data is determined by the maximum significance value [24] which is evaluated in O(k) time where k is the number of level 1 topics and thus can be very effective where time is critical for returning search results.

In Section 2, we present our new hybrid parallel architecture and describe the corpora used to test this architecture. Section 3 details the conversion of text data into the various vector formats and also the classification algorithms used in our experiments. In Section 4, we compare the performance of this hybrid parallel classifier against that of single MLP classifiers using the significance vector as well as the tf-idf vector representation. Our experiments are performed on two different corpora – the Reuters corpus (RCV1) [33] and the Large Scale Hierarchical Text Classification (LSHTC) Corpus [2] using the first two levels of the topic hierarchy in both cases.

2. Methodology overview and overall architecture

The Reuters Corpus is a well-known test bench for text categorization experiments. It also has a hierarchical organization with four major groups which is well suited to test the classification performance of a hybrid architecture. We used the Reuters Corpus headlines for our experiments as they provide a concise summary of each news article. Each Reuters headline consists of one line of text with about 3–12 words. Some example Reuters headlines are given below:

"Healthcare Imaging Q2 loss vs profit."
"Questar signs pact to buy oil, gas reserves."
"Ugandan rebels abduct 300 civilians, army says."
"Estonian president faces reelection challenge."
"Guatemalan sides to sign truce in Norway report."
"CRICKET-Australia beat Zimbabwe by 125 runs in one-day match."
"PRESALE – Akron, Ohio."

The topic codes in the Reuters Corpus represent the subject areas of each news story. They are organized into four hierarchical groups, with four top-level nodes: Corporate/Industrial (CCAT), Economics (ECAT), Government/Social (GCAT) and Markets (MCAT). Under each top-level node there is a hierarchy of codes where the depth of each is represented by the length of the code. As a representative test, ten thousand headlines along with their topic codes were extracted from the Reuters Corpus. These headlines were chosen so that there was no overlap at the first level categorization. Each headline belonged to only one level 1 category. At the second level, since most headlines had multiple level 2 subtopic categorizations, the first subtopic was taken as the assigned subtopic. Thus each headline had two labels associated with it – the main topic (Level 1) label and the subtopic (Level 2) label. Headlines were then preprocessed to separate hyphenated words to avoid such combinations being interpreted as new words rather than a sequence of known words. Dictionaries with term frequencies were generated based on the TMG toolbox [7] and were then used to generate the Full Significance Vector [24], the Conditional Significance Vector [24] and the tf-idf [5] representation for each document. The datasets were then randomised and divided into a training set of 9000 documents and a test set of 1000 documents.

For comparative analysis, we used the LSHTC [2] competition data from the LSHTC website as our second corpus. The LSHTC data has been constructed by crawling the web pages that are found in the Open Directory Project (ODP) located at www.dmoz.org and translating them into feature vectors. These vectors are called content vectors. The Open Directory Project is an open source and extensive directory of web content. An example web page content accessed from this directory is given below:

“Ambienti Italia brings you world class Italian furniture through infinite selections for decorating your home. Flexibility and design expertise allow us to adapt to any kind of space according to required functions and available dimensions. We want our customers to go home and find the best – comfort and style. Ambienti Italia’s collections reflect the achievements and history of Italian home furnishings”

The ODP descriptions of the web pages and the categories are also translated into feature vectors. These vectors are called web page and category description vectors. Two datasets were put up for the LSHTC competition – the large_lshtc_dataset and the smaller dry-run_lshtc_dataset. The directory of each dataset consisted of a *cat_hier.txt* file describing the category hierarchy of the dataset and data folders for four tasks (Task1 – Task4). Task1 contained only crawl data while the data for task 2, task 3 and task 4 contained crawl data and RDF data.

We used the data from the dry-run task1 training folder as our LSHTC corpus. The average number of words in each document in this dataset is 290. This number takes into account only the stemmed words without the stop words. The data is in the form of content vectors which are obtained by directly indexing the web pages. A text file describing the category hierarchy is also given with the data. There were 4463 content vectors in this data file with their associated lowest level labels. We pre-processed these vectors in order to replace the lowest level labels with the corresponding labels of the first two levels of the category hierarchy. These vectors were then used to generate the Full Significance Vector [24], the Conditional Significance Vector [24] and the tf-idf [5] representations for each document as will be described below. The datasets were then randomised and divided into a training set of 4000 vectors and a test set of 463 vectors.

The WEKA machine learning workbench [21] provided various learning algorithms which we combined in various new hybrid architectures in order to test a variety of learning algorithms. Seven algorithms were compared for our representations to examine the performance of various classification algorithms. Classification Accuracy, which is a comparison of the predicted class to the actual class, and the Training Time were recorded for each experiment run.

3. Steps for data processing and data generation for experiments

3.1. Text data preprocessing

For designing and testing our new hybrid architecture, we took text data from two different sources (Reuters and LSHTC). This text data was pre-processed to represent a two-level hierarchy and then processed in a variety of ways to generate data vectors in different formats.

Reuters Corpus: Ten thousand Reuters headlines were used in these experiments. The Level 1 categorization of the Reuters Corpus divides the data into four main topics. These main topics and their distribution in the data along with the number of subtopics of each main topic in this data set are given in Table 1.

Level 2 categorization further divides these into subtopics. Here we took the direct (first level nesting) subtopics of each main topic occurring in the 10,000 headlines. A total of 50 subtopics were included in these experiments. Some of these subtopics with

Table 1
Reuters level 1 topics

No.	Main Topic	Description	Number Present	No. of Subtopics
1.	CCAT	Corporate/ Industrial	4600	18
2.	ECAT	Economics	900	8
3.	GCAT	Government/ Social	1900	20
4.	MCAT	Markets	2600	4

Table 2
Some reuters level 2 subtopics

Main Topic	Subtopic	Description	Number Present
CCAT	C17	Funding/ Capital	377
CCAT	C32	Advertising/ Promotion	10
ECAT	E12	Monetary/ Economic	107
ECAT	E21	Government Finance	377
GCAT	G15	European Community	38
GCAT	GENV	Environment	30
MCAT	M11	Equity Markets	617
MCAT	M14	Commodity Markets	1050

Table 3
LSHTC level 1 (main) topics

No.	Main Topic	Number Present	Number of Subtopics
1.	A	802	19
2.	B	979	36
3.	C	639	17
4.	D	269	17
5.	E	158	5
6.	F	20	3
7.	G	578	19
8.	H	364	6
9.	I	321	14
10.	J	333	22

their numbers present are shown in Table 2. Since all the headlines had multiple subtopic assignments, e.g. C11/C15/C18, only the first subtopic e.g. C11 was taken as the assigned subtopic. Our assumption here is that the first subtopic used to tag a particular Reuters news item is the one which is most relevant to it.

LSHTC Corpus: This dataset consisted of 4463 content vectors with multilevel categorization. There was no data with overlapping categorization in this dataset. There are 10 level 1 and 158 level 2 topics in this corpus. These topics were coded numerically. We replaced this numeric code with an alphanumeric code for ease of analysis. Subsequently the 10 top level categories were given letter codes A – J. These main topics and their distribution in the data along with the number of subtopics of each main topic in this data set are given in Table 3. The subtopics were coded A01-A19, B01-B36, etc. with the first character denoting the main topic to which these subtopics belonged. The number of data vectors for some of these subtopics is given in Table 4.

Table 4
Some LSHTC level 2 subtopics

Subtopic	Number Present	Subtopic	Number Present
A09	120	F02	11
A16	8	F03	8
B06	114	G07	47
B26	40	G14	208
C05	2	H02	336
C10	232	H04	2
D02	26	I03	91
D08	62	I10	18
E03	40	J06	44
E05	2	J22	19

3.2. Semantic significance vector generation

We use a vector representation which represents the significance of the data and weighs different words according to their significance for different topics. Significance Vectors [30,31] are determined based on the frequency of a word in different semantic categories. A modification of the significance vector called the semantic vector uses normalized frequencies where each word w is represented with a vector (c_1, c_2,...,c_n) where c_i represents a certain semantic category and n is the total number of categories. A value v(w, c_i) is calculated for each element of the semantic vector as the normalized frequency of occurrences of word w in semantic category c_i (the normalized category frequency), divided by the normalized frequency of occurrences of the word w in the corpus (the normalized corpus frequency):

$$\text{v}(\text{w}, c_i) = \frac{\text{Normalised Frequency of w in } c_i}{\sum_k \text{Normalised Frequency of w in } c_k}$$

$$\text{where } k \in \{1..\text{n}\}$$

For each document, the document semantic vector is obtained by summing the semantic vectors for each word in the document and dividing by the total number of words in the document. Henceforth it is simply referred to as the *Significance Vector*. The TMG Toolbox [7] was used to generate the term frequencies for each word in each headline. The word vector consisted of 54 columns (for 4 main topics and 50 subtopics) for the Reuters Corpus and 168 columns (for 10 main topics and 158 subtopics) for the LSHTC corpus. While calculating the significance vector entries for each word, its occurrence in all subtopics of all main topics was taken into account – hence called the *Full Significance Vector*. We also generate the *Conditional Significance Vector* [24] where a word's occurrence in all subtopics *of only a particular main topic* is taken into account while calculating the word significance vector entries.

3.3. *Data vector sets generation*

As will be described below, three different vector representations (Full Significance Vector, Conditional Significance Vector and tf-idf) were generated for our data. The Conditional Significance Vectors were processed further to generate main category-wise data vector sets (4 different datasets for Reuters and 10 different data sets for LSHTC).

3.3.1. *Full significance vector*

Here, the document vectors were generated by summing the full significance word vectors for each word occurring in a document and then dividing by the total number of words in that document. For each Reuters Full Significance document vector the first four columns, representing four main topics – CCAT, ECAT, GCAT & MCAT, were ignored leaving a vector with 50 columns representing 50 subtopics. The order of the data vectors was then randomised and divided into two sets – training set of 9000 vectors and a test set of 1000 vectors. Similarly, for each LSHTC Full Significance document vector the first ten columns, representing ten main topics (A–J), were ignored leaving a vector with 158 columns representing 158 subtopics. The order of the data vectors was then randomised and divided into two sets – training set of 4000 vectors and a test set of 463 vectors.

3.3.2. *Category-based conditional significance vectors*

Here, the conditional significance word vectors were used to generate the document vectors in the same way as above for the Reuters and LSHTC corpora. These document vectors were then processed as described below to produce the *CSV_RelVectors* for each corpus.

Reuters Corpus: The order of the 10,000 Reuters Conditional Significance document vectors was randomised and divided into two sets – a training set of 9000 vectors and a test set of 1000 vectors. The training set was then divided into 4 sets according to the main topic labels. For each of these sets, only the relevant subtopic vector entries (e.g. C11, C12, etc. for CCAT; E11, E12, etc. for ECAT) for each main topic were retained. Thus the CCAT category training dataset had 18 columns for 18 subtopics of CCAT. Similarly the ECAT training dataset had 8 columns, the GCAT training dataset had 20 columns and the MCAT training dataset had 4 columns. These 4 training sets were then used to train the 4 parallel classifiers of the Reuters hybrid classifier. The main category of a test data vector was determined by the maximum significance vector entry for the first four columns representing the four main categories. After this, the entries corresponding to the subtopics of this predicted main topic were extracted along with the *actual* subtopic label and given to the classifier trained for this predicted main category.

LSHTC Corpus: The order of the 4463 LSHTC Conditional Significance document vectors was randomised and divided into two sets – training set of 4000 vectors and a test set of 463 vectors. The training set was then divided into 10 sets according to the main topic labels. For each of these for sets, only the relevant subtopic vector entries (e.g. A01, A02, etc. for A; B01, B02, etc. for B) for each main topic were retained. These 10 training sets were then used to train the 10 parallel classifiers of the LSHTC hybrid classifier. The main category of a test data vector was determined by the maximum significance vector entry for the first ten columns representing the ten main categories. After this, the entries corresponding to the subtopics of this predicted main topic were extracted along with the *actual* subtopic label and given to the classifier trained for this predicted main category.

Figure 3 shows the classification decisions for some Reuters input vectors. Figures 3(a)–3(e) each represent one input test vector. The x-axis of these figures represents the significance vector components which in turn represent all the main topics and subtopics present in our Reuters Corpus data. The y-axis shows the actual numerical values for these significance vector components as calculated in Sections 3.2 and 3.3. The black data points show the predicted main topic and the predicted subtopic while the gray data points show the actual main topic and the actual subtopic (wherever actual and predicted are distinct). Figures 3(a), 3(b) and 3(c) show correctly classified vectors while Figures 3(d) and 3(e) show vectors which are classified wrongly. In Figures 3(a), 3(b) and 3(c), there are no gray data points as the predicted and actual main topics are the same. In Fig. 3(d), the main topic predicted was correct and the vector was presented to the correct classifier but the subtopic classification was wrong. Hence the figure shows black and gray data points for the subtopic. In Fig. 3(e), the main topic predicted was wrong and hence the vector was presented to the wrong classifier – resulting in a wrong classification. This figure shows black and gray data points for both the main topic as well as the subtopic. Figure 3(e) presents an inherent limitation of this system whereby a wrong classifier is chosen by the classifier selection step of the parallel classifier.

For the Reuters Corpus, the accuracy of choosing the correct main topic by selecting the maximum significance level 1 entry was measured to be 96.80% for the 1000 test vectors, i.e. 968 vectors were assigned the correct trained classifiers whereas 3.20% or 32 vectors were assigned to a wrong classifier – resulting in a wrong classification decision for all these 32 vectors. Hence the upper limit for classification accuracy is 96.80% for our hybrid parallel classifier for the Reuters Corpus. Similarly, the accuracy of choosing the correct main topic by selecting the maximum significance level 1 entry was measured to be 85.31% for the 463 LSHTC test vectors, i.e. 85.31% or 395 vectors were assigned the correct trained classifiers whereas 14.69% or 68 vectors were assigned to a wrong classifier – resulting in a wrong classification decision for all these 68 vectors. Hence the upper limit for classification accuracy is 85.31% for our hybrid parallel classifier for the LSHTC Corpus. Figures 4(a), 4(b) and 4(c) show relevant snapshots of the correctly classified LSHTC vectors while Figs 4(d) and 4(e) show snapshots of the LSHTC vectors which are classified wrongly.

3.3.3. Category-based full significance vectors

To compare the performance of different vector formats, we also generated the category-based Full Significance Vectors. Here, the Full Significance document vectors were generated as described in Section 3.3.1 for the Reuters and LSHTC Corpora. After this, the document vector set for each corpus was divided into category-based training and test sets as described in section 3.3.2.

Two variations of the category based Full Significance Vectors were generated for our experiments:

i) Category-Wise Separated Vectors with the complete set of subtopic vector dimensions (50 for Reuters and 158 for LSHTC) designated as *FSV_FullVector;*
ii) Category-Wise Separated Vectors with only the relevant subtopic vector dimensions corresponding to the actual main category for training vectors and the predicted main category for test vectors. These vectors are designated here as *FSV_RelVector*.

3.3.4. TF-IDF vector generation

The tf-idf weight (Term Frequency–Inverse Document Frequency) is often used in text mining and information retrieval. It is a statistical measure which evaluates how important a word is to a document in a data set. This importance increases with the number of times a word appears in the document but is reduced by the frequency of the word in the data set. Words which occur in almost all documents have very little discriminatory power and hence are given very low weight. The TMG toolbox [7] was used to generate the tf-idf vectors for our experiments. The tf-idf vector datasets were then randomized and divided into 9000 training vectors / 1000 test vectors for the Reuters Corpus and 4000 training vectors / 463 test vectors for the LSHTC Corpus.

3.4. Classification algorithms

Seven Classification algorithms were tested with our datasets namely Random Forest, C4.5, the Multilayer Perceptron, Naïve Bayes, BayesNet, NNge and PART. Random Forests [16,27] are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently. C4.5 [12,29] is an inductive tree algorithm with two pruning methods: subtree replacement and subtree raising. The Multilayer Perceptron [4,22] is a neural network which uses backpropagation for training. Naive Bayes [10,17] is the simplest form of Bayesian network, in which all attributes are independent given the value of the class variable. BayesNet [9,18] implements Bayes Network learning using various search algorithms and quality measures. NNge [3] is a nearest neighbor - like algorithm using non-nested generalized exemplars which can be considered as if-then rules. A PART [8] decision list uses C4.5 decision trees to generate rules. Table 5 shows the different classification algorithms used with their default parameters in Weka.

4. Results and their analysis

A variety of basic learning algorithms required to test various hybrid combinations for our new architecture were provided by the WEKA machine learning workbench [21]. The Multilayer Perceptron (MLP) along with six other basic algorithms were used in our experiments. These included two Bayesian algorithms (BayesNet and Naive Bayes), two rule-based al-

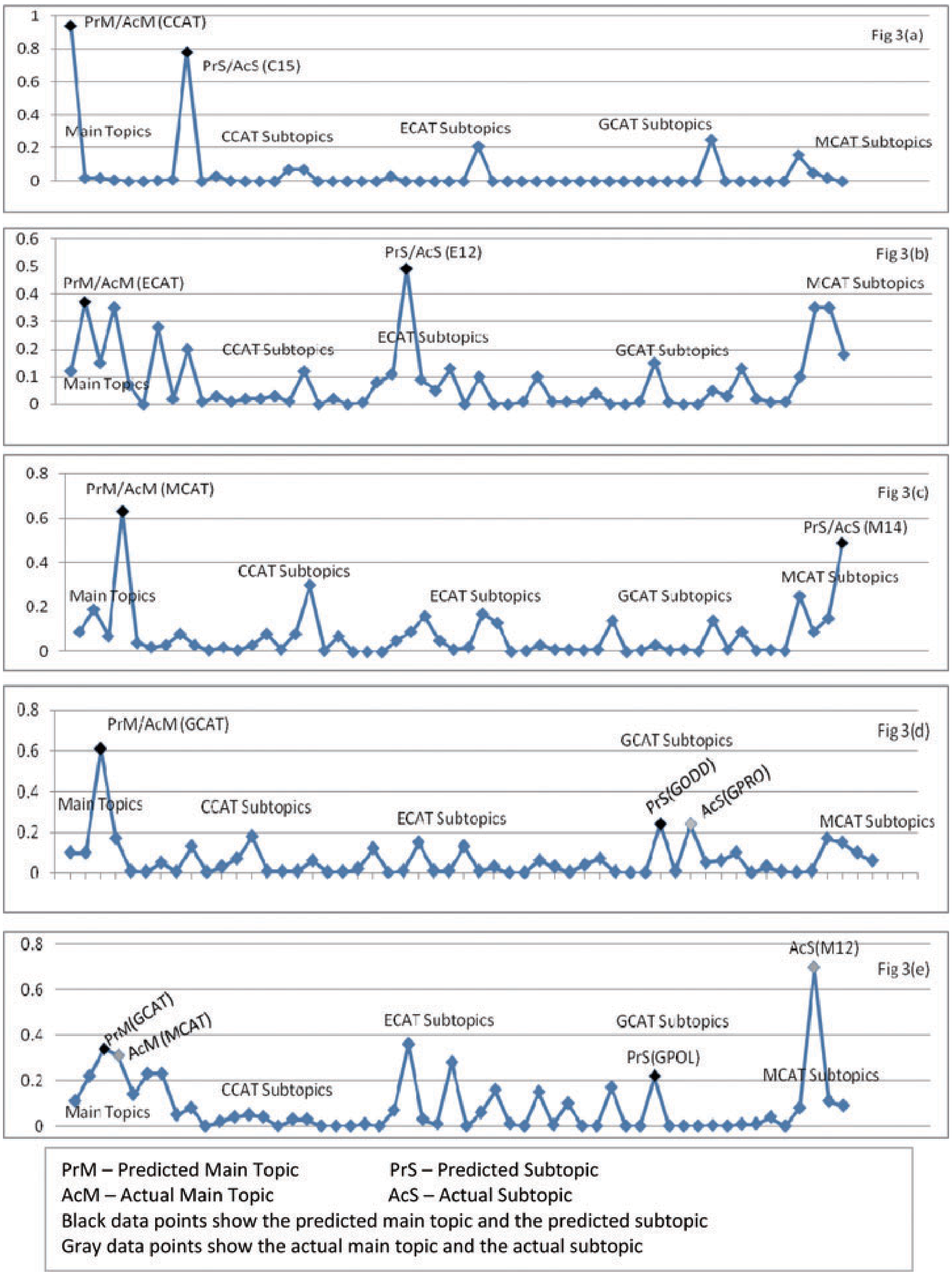


Fig. 3. Classification decisions by a hybrid parallel classifier for some REUTERS input vectors.

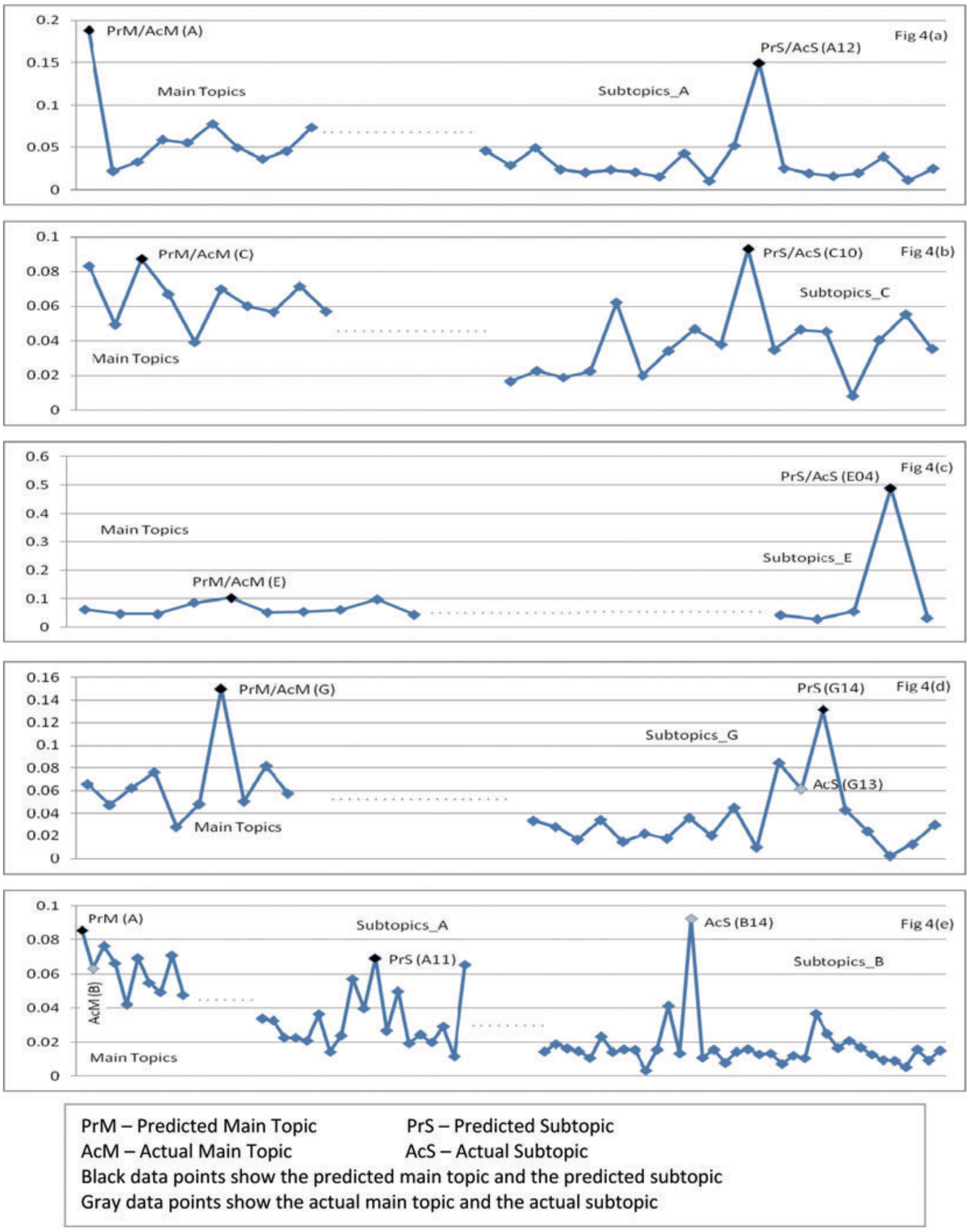


Fig. 4. Classification decisions by a hybrid parallel classifier for some LSHTC input vectors.

Table 5
Classification algorithms and their default settings in weka

No.	Algorithm	Default settings
1.	BayesNet	Estimates probabilities directly from the data; Uses the K2 hill climbing algorithm;
2.	Naïve Bayes	Numeric estimator precision values are chosen based on analysis of the training data;
3.	PART	Confidence factor for pruning = 0.25; Minimum Number of instances per rule = 2;
4.	NNge	Number of Attempts for Generalisation = 5; Number of Folders for Mutual Information = 5;
5.	J48(C4.5)	Confidence factor for pruning = 0.25, Minimum Number of Instances per leaf = 2; Subtree raising used on pruning;
6.	Random Forest	Number of Trees to be generated = 10; No limit on the depth of a tree;
7.	Multilayer Perceptron	Number of hidden layers = (attributes + classes) / 2; Learning Rate = 0.3; Momentum = 0.2; Training Time = 500; Validation threshold = 20;

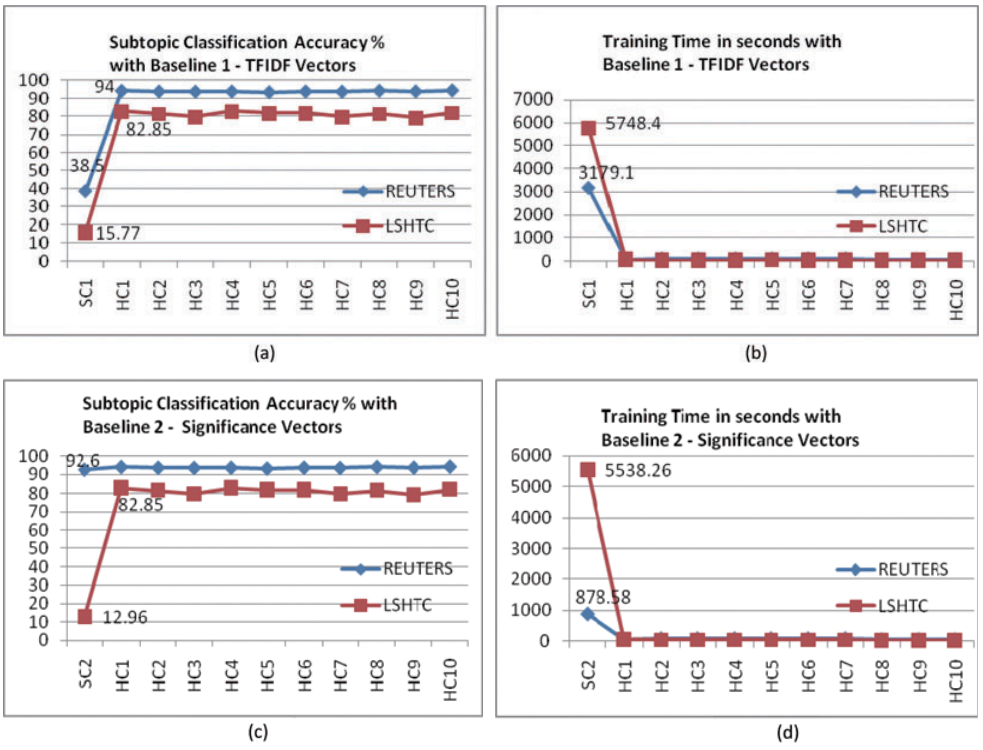
gorithms (PART and NNge) and two tree-based algorithms (J48 and Random Forest).

Our experiments were run using these seven algorithms from Weka on the Reuters and LSHTC Corpora. The Reuters Corpus was divided into 9000 training vectors and 1000 test vectors while the LSHTC Corpus was divided into 4000 training and 463 test vectors. For the hybrid classifier, the 9000 training vectors for Reuters and the 4000 training vectors for LSHTC were divided according to the actual main categories and were used to train the chosen category classifier with the relevant subtopic vector entries and actual subtopic labels. The test vectors were divided into main categories based on the maximum significance value among the main topic significance vector entries. The relevant subtopic vector entries of this predicted main topic and the *actual* subtopic labels of these vectors were used to test these classifiers.

In the first step, we used the category-wise separated data from the training set to select the algorithm with the highest classification accuracy for each main category. In the case of a tie between two algorithms, the one with the lower training time was chosen. Subsequently we applied these selected algorithms to the test data and measured the performance of the hybrid classifier. The category-wise separated Conditional Significance Vectors were used here. We also ran each of the basic algorithms on the full (not category-wise separated) data set to provide a comparison for the hybrid classifier. Two vector representations were used for the comparison baseline – the Full Significance Vector and tf-idf. As the performance of many classifiers for each main category was quite close to each other, we also ran some experiments using a predefined set of classifiers. The combination of MLP with different types of classifiers (Bayesian, rule-based and tree-based classifiers) was evaluated and the best combination was identified. For a two-classifier combination, MLP and the other classifier were used alternately on the main category topics while for a four-classifier system four different classifiers were used on the four main topics of Reuters Corpus and repeated for each block of four main topics for the LSHTC Corpus.

The charts in Fig. 5 show a comparison of the performance of hybrid classifiers with that of MLP for both corpora. The subtopic classification accuracy percentage and training time in seconds is shown for the Hybrid Parallel classifiers along with that of the baselines. The baseline is a single MLP classifier using full data (not category-wise separated data). This baseline experiment is run with two different vector representations – Significance Vector and tf-idf. The accuracies of all the hybrid parallel classifiers are better than that of the single MLP classifier. This could be due to the fact that each base classifier present in the hybrid parallel classifier has to learn from a subset of the original data. As such, it is able to distinguish between categories present in this subspace more accurately than a classifier which has to learn from the full dataset.

Overall, it was observed that there was an improvement in subtopic classification accuracy along with a significant reduction in training time. The classification accuracies of all the hybrid classifiers were quite close to each other but all of them were much better than the classification accuracy of the single classifier with tf-idf baseline for both the Reuters and the LSHTC corpora. The difference with the significance vector baseline was smaller for the Reuters Corpus but even there the classification accuracies of the hybrid systems were better. The training times showed a very steep



Classifier Index

SC1- Single MLP over full data using tf-idf Vectors
SC2- Single MLP over full data using Significance Vectors
HC1- Hybrid Parallel Classifier with category-wise classifiers chosen from training data performance
HC2- Hybrid 2-Classifier System (MLP/NB)*
HC3- Hybrid 2-Classifier System (MLP/BNet)*
HC4- Hybrid 2-Classifier System (MLP/NNge)*
HC5- Hybrid 2-Classifier System (MLP/PART)*
HC6- Hybrid 2-Classifier System (MLP/J48)*
HC7- Hybrid 2-Classifier System (MLP/RF)*
HC8- Hybrid 4-Classifier System (MLP/NB/ NNge/ J48)*
HC9- Hybrid 4-Classifier System (MLP/BNet/PART/RF)*
HC10- Hybrid 4-Classifier System (MLP/NNge/PART/NB)*

*MLP - Multilayer Perceptron (Neural Network); NB - Naïve Bayes, BNet - BayesNet (Bayesian);
NNge - Nearest Neighbour with Generalised Exemplars, PART - PART Decision List (Rule Based);
J48 - Weka's version of C4.5, RF - Random Forest (Tree Based);

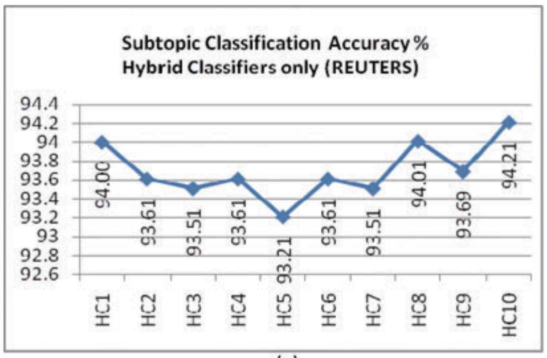
Fig. 5. Hybrid parallel classifiers performance metrics with baselines.

reduction compared to both baselines. The average of 10 runs was taken for each experiment. In the hybrid classifier, even though we are using more classifiers, the training time is reduced. This is because each classifier now trains on a reduced set of data with a reduced set of vector components. This two-fold reduction translates to a significant decrease in training time.

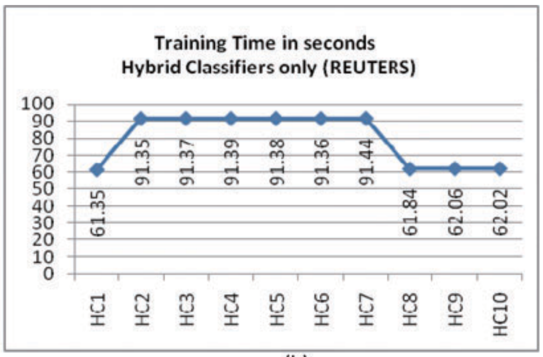
We also compared the performance of one hybrid classifier (HC4) with three different vector formats: FSV_FullVector, FSV_RelVector and CSV_RelVector. It was observed that the CSV_RelVector gave the best subtopic classification accuracy.

4.1. Reuters corpus results

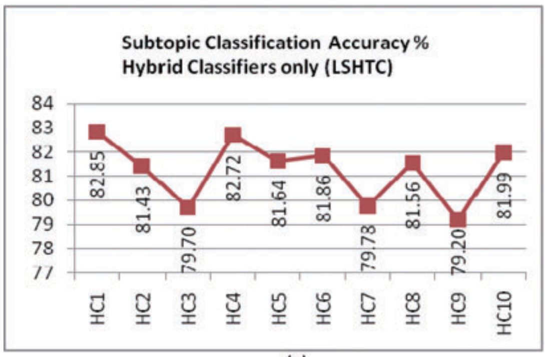
Figures 6(a) and 6(b) show the detailed results for the Reuters Corpus. The Hybrid 4-classifier system (HC10) shows the best classification accuracy



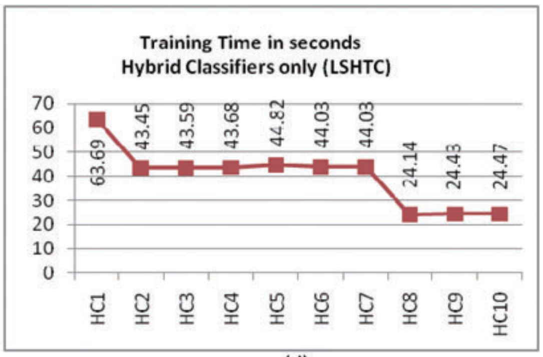
(a)



(b)



(c)



(d)

Classifier Index

HC1- Hybrid Parallel Classifier with category-wise classifiers chosen from training data performance
HC2- Hybrid 2-Classifier System (MLP/NB)*
HC3- Hybrid 2-Classifier System (MLP/BNet)*
HC4- Hybrid 2-Classifier System (MLP/NNge)*
HC5- Hybrid 2-Classifier System (MLP/PART)*
HC6- Hybrid 2-Classifier System (MLP/J48)*
HC7- Hybrid 2-Classifier System (MLP/RF)*
HC8- Hybrid 4-Classifier System (MLP/NB/ NNge/ J48)*
HC9- Hybrid 4-Classifier System (MLP/BNet/PART/RF)*
HC10- Hybrid 4-Classifier System (MLP/NNge/PART/NB)*

*MLP - Multilayer Perceptron (Neural Network); NB - Naïve Bayes, BNet - BayesNet (Bayesian);
NNge - Nearest Neighbour with Generalised Exemplars, PART - PART Decision List (Rule Based);
J48 - Weka's version of C4.5, RF - Random Forest (Tree Based);

Fig. 6. Hybrid parallel classifiers only - Performance metrics.

which is quite similar to that of the hybrid classifier with category-wise classifiers chosen from training set (HC1). The training times of all hybrid classifiers were quite close to each other with HC1, HC8, HC9 and HC10 showing the least training time. The other hybrid classifiers were two-classifier systems with one MLP and one non-MLP classifier alternating on the main topics. Hence for the Reuters data with four main topics, there were two MLPs in all the hybrid 2-classifier systems. This could account for the slightly higher training time of these classifiers versus the hybrid 4-classifier systems (HC8, HC9 and HC10) which have only one MLP in the combination. The hybrid classifier with category-wise classifiers chosen from training

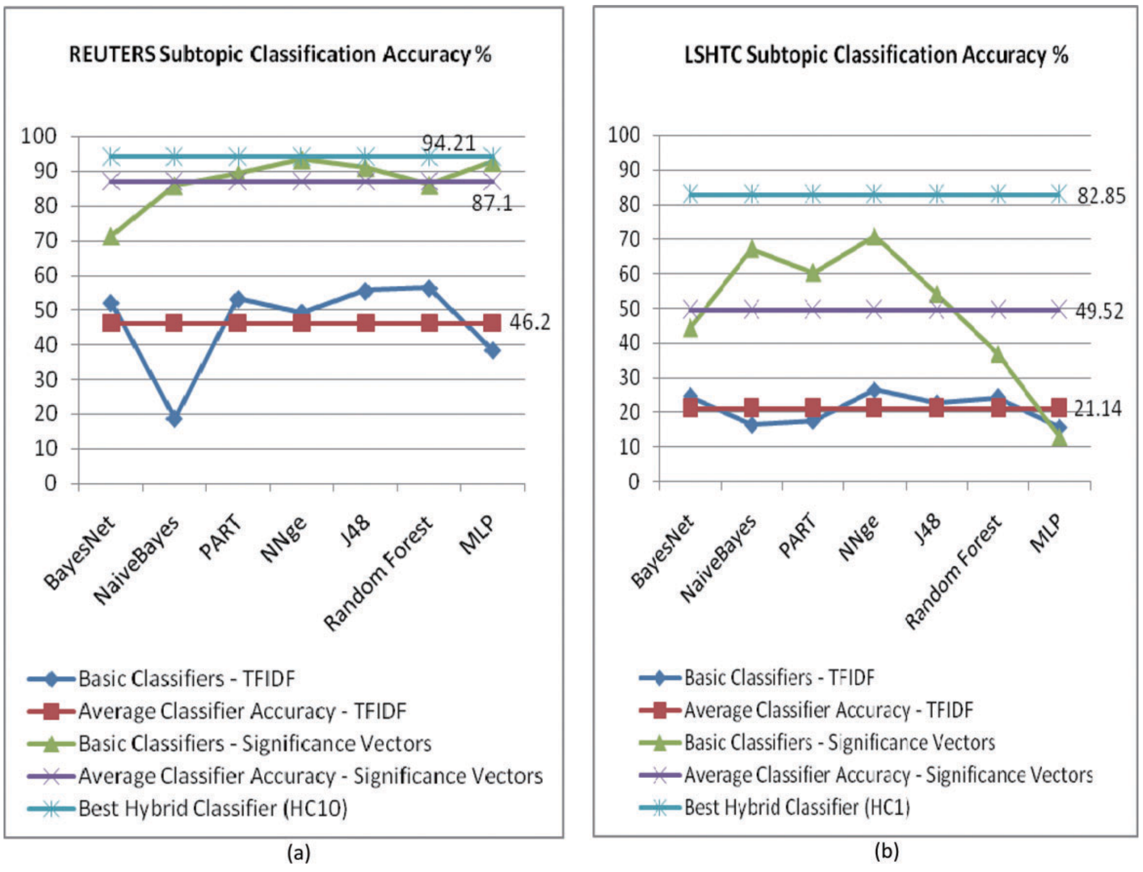


Fig. 7. Comparison of hybrid classifier performance with basic classifiers on full data.

set (HC1) had MLP for the CCAT main topic and J48 for all other main topics. Since this combination also had only one MLP, its training time was comparable to the hybrid 4-classifier systems.

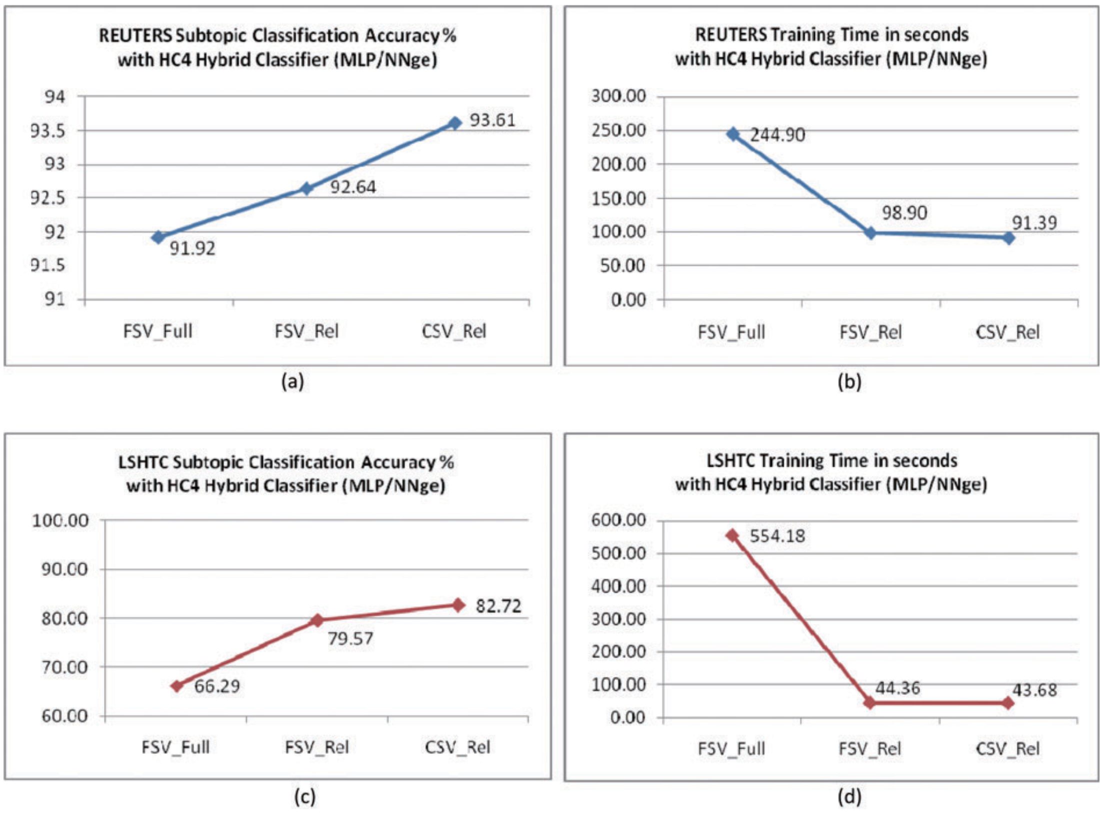
Figure 7(a) shows the comparison of the classification accuracy of the best hybrid classifier (HC10) on category-wise data with that of each basic classifier on full data. The average classification accuracy is also shown. The chart shows the performance of each basic classifier using two different vector formats – tf-idf and Significance Vector. The performance of the hybrid classifier is better than the average basic classifier accuracy for both vector formats.

Figures 8(a) and 8(b) shows the performance of the HC4 classifier (Hybrid parallel 2-classifier MLP/NNge combination) with different vector formats for the Reuters Corpus. It can be seen that CSV_RelVector (Conditional Significance Vectors with only the relevant subtopic vector components) gives the highest subtopic classification accuracy and the lowest training time.

4.2. LSHTC corpus results

Figures 6(c) and 6(d) show the detailed results for the LSHTC Corpus. The highest subtopic classification accuracy is shown by the Hybrid Parallel Classifier with category-wise classifiers chosen from training data performance (HC1) with 82.85%. It has a training time of 63.69 seconds. This is very closely followed by Hybrid 2-Classifier (MLP/NNge) System (HC4) with 82.72% classification accuracy and 43.68 seconds training time. The lowest training time is shown by the Predefined Hybrid 4-Classifier System (MLP/NB/NNge/J48) (HC8) at 24.14 seconds. In an overall tradeoff between classification accuracy and training time, the best hybrid classifier seems to be the Hybrid 2-Classifier System (MLP/NNge) (HC4). This classifier also eliminates the step of choosing the best classifier per main category from the training set and thus effectively reduces training time even further.

Figure 7(b) shows the comparison of the classification accuracy of the best hybrid classifier (HC1) on



HC4: Hybrid Parallel 2-Classifier Combination with MLP and NNge

Vector Formats:

FSV_Full: Full Significance Vector with the full set of subtopic vector components (Vector Length is 50 for Reuters and 158 for LSHTC)
FSV_Rel : Full Significance Vector with only the set of subtopic vector components relevant to the Main Topic
CSV_Rel : Conditional Significance Vector with only the set of subtopic vector components relevant to the Main Topic

Fig. 8. Comparison of hybrid classifier (HC4) performance with different vector formats.

category-wise data with that of each basic classifier on full data for the LSHTC Corpus. The average classification accuracy is also shown. The chart shows the performance of each basic classifier using two different vector formats – tf-idf and Significance Vector. The performance of the hybrid classifier is much better than the average basic classifier accuracy for both vector formats.

Figures 8(c) and 8(d) show the performance of the HC4 classifier (Hybrid parallel 2-classifier MLP/NNge combination with different vector formats for the LSHTC Corpus. Here again, it can be seen that CSV_RelVector (Conditional Significance Vectors with only the relevant subtopic vector components) gives the best subtopic classification accuracy and training time. The improvement is higher with the LSHTC Corpus than with the Reuters Corpus.

The classification accuracy of the hybrid classifier is better than the average basic classifier accuracy for both vector formats. The improvement in performance is much more marked with the LSHTC Corpus as compared to the Reuters Corpus. As the LSHTC Corpus has more categories (10 main and 158 subtopic) than the

Reuters Corpus (4 main and 50 subtopics), this result is particularly encouraging.

5. Conclusion

In this paper, we attempt to leverage the differences in the characteristics of different subspaces to improve semantic subspace learning. The main objective here is to improve document classification in a document space by combining various learning methods. Our experiments show that hybrid parallel combinations of classifiers trained on different subspaces offer a significant performance improvement over single classifier learning on full data space. Individual classifiers also perform better when presented with less data in lower dimensions. Our experiments also show that learning based on the semantic separation of the data space is more efficient than full data space learning. Combining different types of classifiers has the advantage of integrating characteristics of different subspaces and hence improves classification performance. Future work should test whether this approach can work well in other domains like pattern / image recognition where different classifiers can work on different parts of the image to improve overall recognition.

In our experiments, subspace detection is done by processing a single document vector. This method is independent of the total number of data samples and only compares the level 1 topic entries. The time complexity of the combining classifier is thus O(k) where k is the number of level 1 topics. The novelty of our approach is in the use of a maximum significance based method of input vector projection for a hybrid parallel classifier. Combining MLP in parallel with a basic classifier (Bayesian, tree based or rule based) improves the classification accuracy and significantly reduces the training time. The performance improvement is even more significant when the number of topics and subtopics is large (LSHTC v/s Reuters). The experiments also show that using the maximum significance value is very effective in detecting the relevant subspace of a test vector and that conditional significance vectors further boost subtopic classification accuracy.

References

[1] A. Estabrooks and N. Japkowicz, A mixture-of-experts framework for text classification, In Proceedings of the 2001 Workshop on Computational Natural Language Learning – Volume 7 (Toulouse, France, July 06–07, 2001). pp. 1–8.

[2] A. Kosmopoulos, E. Gaussier, G. Paliouras and Aseervatham, The ECIR 2010 Large Scale Hierarchical Classification Workshop, In SIGIR Forum, June 2010, Volume 44 Number 1, 2010, pp. 23–32.

[3] B. Martin, Instance-Based learning: Nearest Neighbor With Generalization, Master Thesis, University of Waikato, Hamilton, New Zealand, 1995.

[4] B. Verma, Fast training of multilayer perceptrons, *IEEE Transactions on Neural Networks* **8**(6) (Nov 1997), 1314–1320.

[5] C. Manning, P. Raghavan and H. Schutze, *Introduction to Information Retrieval*, Cambridge University Press, 2008.

[6] D. Fradkin and D. Madigan, Experiments with Random Projections for Machine Learning, In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003, pp. 517–522.

[7] D. Zeimpekis and E. Gallopoulos, TMG: A MATLAB Toolbox for Generating Term Document Matrices from Text Collections, in: *Book Chapter in Grouping Multidimensional Data: Recent Advances in Clustering*, J. Kogan and C. Nicholas, eds, Springer, 2005.

[8] E. Frank and I.H. Witten, Generating Accurate Rule Sets Without Global Optimization, in: *Machine Learning: Proceedings of the Fifteenth International Conference*, J. Shavlik, ed., Morgan Kaufmann Publishers, 1998.

[9] F. Pernkopf, *Discriminative Learning of Bayesian Network Classifiers*, In Proceedings of the 25th IASTED International Multi-Conference: artificial intelligence and applications, 2007, pp. 422–427.

[10] H. Zhang and J. Su, Naive Bayes for Optimal ranking, *Journal of Experimental and Theoretical Artificial Intelligence* **20**(2) (June 2008), 79–93.

[11] J.H. Friedman, On Bias, Variance, 0/1–Loss, and the Curse-of- Dimensionality, *In Data Mining and Knowledge Discovery* **1**(1) (1997), 55–77.

[12] J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[13] K. Al-Kofahi, A. Tyrrell, A. Vachher, T. Travers and P. Jackson, *Combining Multiple Classifiers for Text categorization*, Proceedings of the tenth international conference on Information and knowledge management, CIKM 2001, pp. 97–104.

[14] K.R. Varshney and A.S. Willsky, *Learning Dimensionality-Reduced Classifiers for Information Fusion*, In Proceedings of the 12th International Conference on Information Fusion, pages 1881–1888, Seattle, Washington, July 2009.

[15] L. Breiman, Bagging predictors, *Machine Learning* **24**(2) (1996), 123–140.

[16] L. Breiman, Random Forests, *Machine Learning* **45**(1) (Oct 2001), 5–32.

[17] L. Jiang, D. Wang, Z. Cai and X. Yan, *Survey of Improving Naïve Bayes for Classification*, Proceedings of the 3^{rd} International Conference on Advanced Data Mining and Applications (ADMA '07), 2007, pp. 134–145.

[18] L. Likforman-Sulem and M. Sigelle, Recognition of degraded characters using dynamic Bayesian networks, *Pattern Recognition* **41**(10) (October 2008), 3092–3103.

[19] L. Parsons, E. Haque and H. Liu, Subspace Clustering for High Dimensional Data : A Review, *ACM SIGKDD Explorations Newsletter* **6**(1) (2004), 90–105.

[20] M.G. Ruiz and P. Srinivasan, Hierarchical Neural Networks for Text Categorization, SIGIR 1999.

[21] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. Witten, The WEKA Data Mining Software: An Update, *ACM SIGKDD Explorations Newsletter* **11**(1) (July 2009), 10–18.

[22] M. Popescu, V. Balas, L. Perescu-Popescu and N. Mastorakis, MultiLayer Perceptron and Neural Networks, *WSEAS Transactions on Circuits and Systems* **8**(7) (July 2009), 579–588.

[23] N. Garcia-Pedrajas and D. Ortiz-Boyer, Boosting Random Subspace Method, *Neural Networks* **21** (2008), 1344–1362.

[24] N. Tripathi, S. Wermter, C. Hung and M. Oakes, *Semantic Subspace Learning with Conditional Significance Vectors*, Proceedings of the IEEE International Joint Conference on Neural Networks, pp. 3670–3677, Barcelona, July 2010

[25] R.E. Schapire, The boosting approach to machine learning: An overview, Nonlinear Estimation and Classification, *Lecture Notes in Statist* **171** (2003), 149–171. Springer, New York.

[26] R.P.W. Duin and D.M.J. Tax, in: *Experiments with Classifier Combining Rules*, J. Kittler and F. Roli, eds, MCS 2000, LNCS 1857, 2000, pp. 16–29.

[27] S. Bernard, L. Heutte and S. Adam, *On the Selection of Decision Trees in Random Forests,* Proceedings of the International Joint Conference on Neural Networks (IJCNN '09) , 2009, pp. 790–795.

[28] S.B. Kotsiantis, *Local Random Subspace Method for Constructing Multiple Decision Stumps*, International Conference on Information and Financial Engineering, 2009, pp. 125–129.

[29] S. Ruggieri, Efficient C4.5, *IEEE Transactions on Knowledge and Data Engineering* **14**(2) (March 2002), 438–444.

[30] S. Wermter, C. Panchev and G. Arevian, *Hybrid Neural Plausibility Networks for News Agents*, In Proceedings of the Sixteenth National Conference on Artificial Intelligence, 1999, pp. 93–98.

[31] S. Wermter, *Hybrid Connectionist Natural Language Processing*, Chapman and Hall, 1995.

[32] Tin Kam Ho, The random subspace method for constructing decision forests, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**(8) (Aug 1998), 832–844.

[33] T. Rose, M. Stevenson and M. Whitehead, The Reuters Corpus Volume 1 – from Yesterday's News to Tomorrow's Language Resources, In Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-02), 2002, pp. 827–833.

[34] Y. Yaslan and Z. Cataltepe, Co-training with relevant random subspaces, *Neurocomputing* **73** (2010), 1652–1661 (Elsevier).