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.

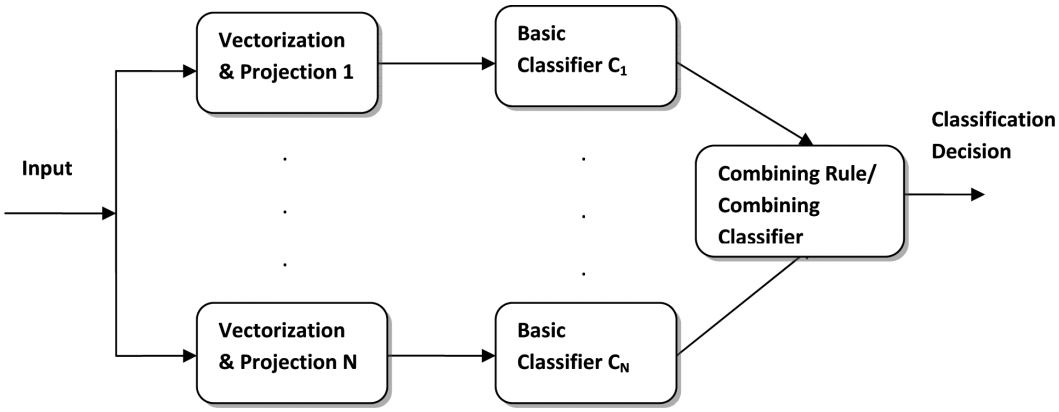

Fig. 1. A combined classifier.

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-

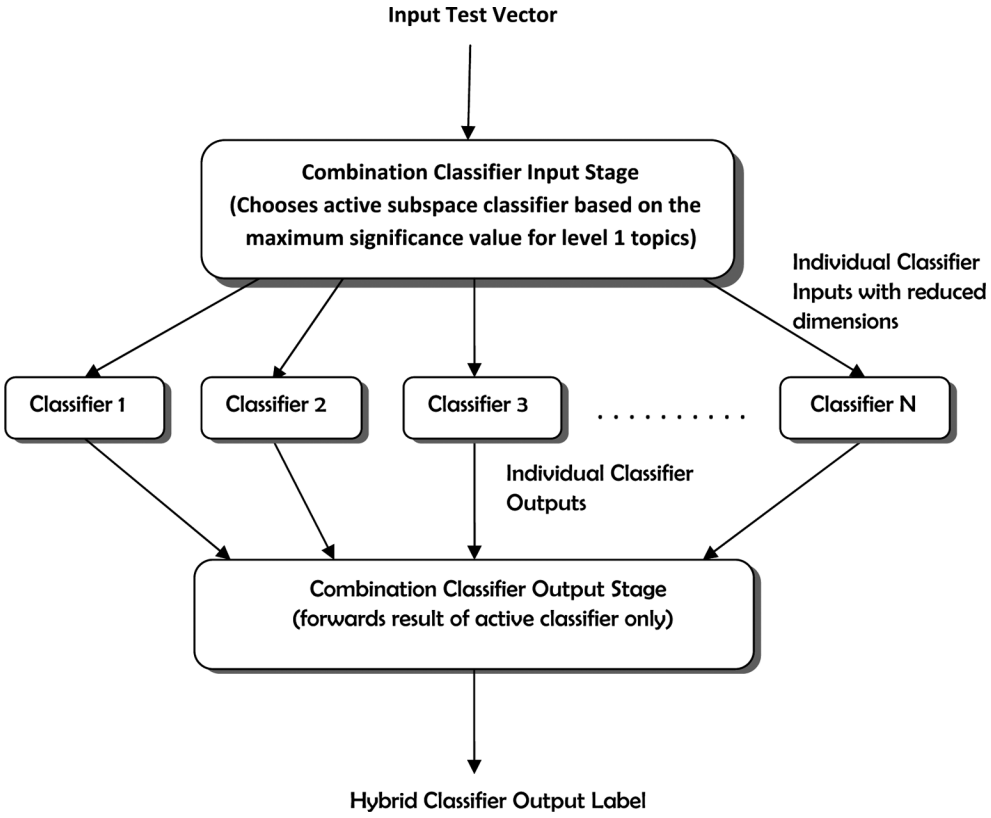

Fig. 2. Hybrid parallel classifier architecture for subspace learning.

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 hierar-

chy 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):

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

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

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 fig-

ure 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-

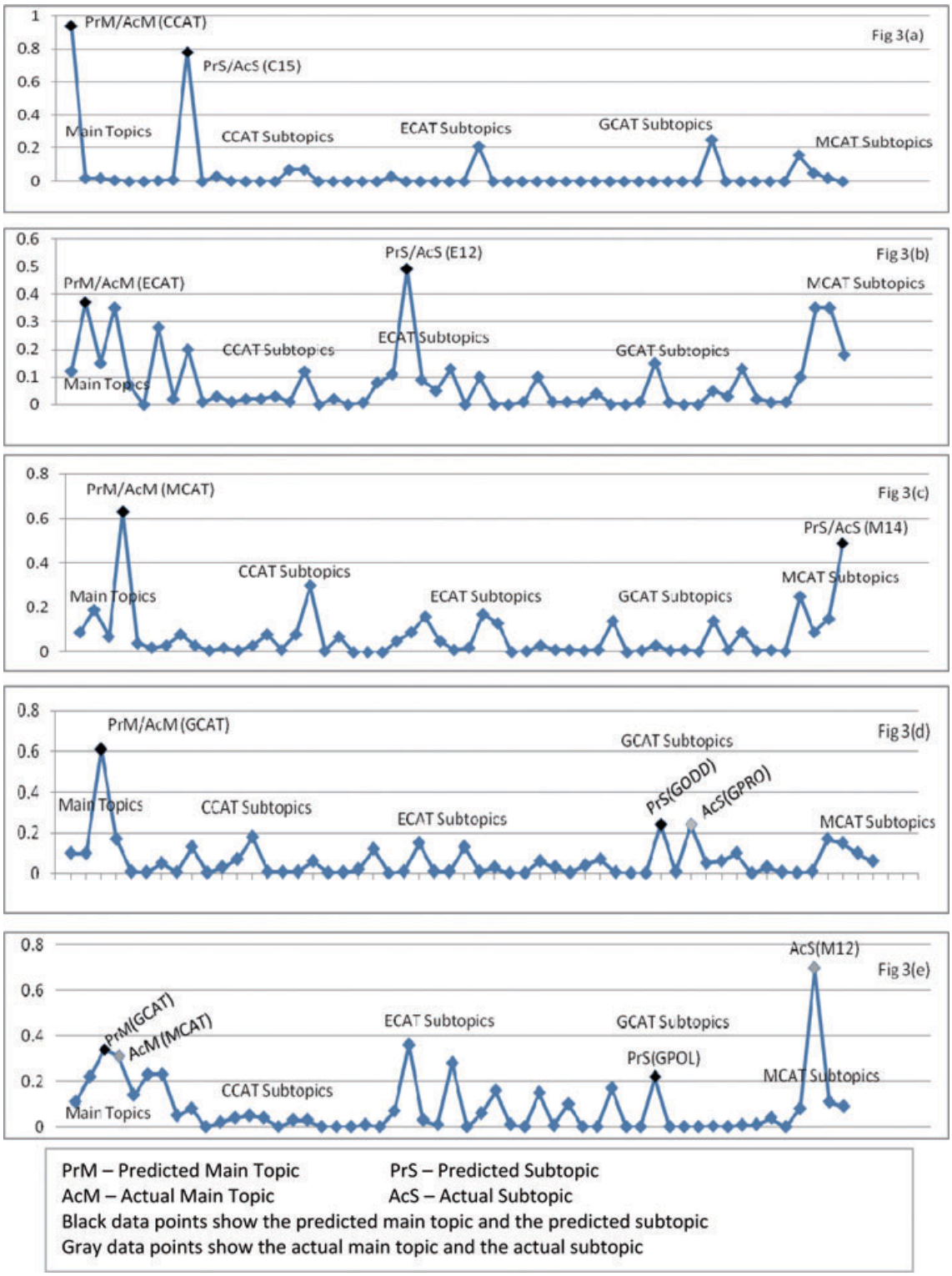

Fig. 3. Classification decisions by a hybrid parallel classifier for some REUTERS input vectors.

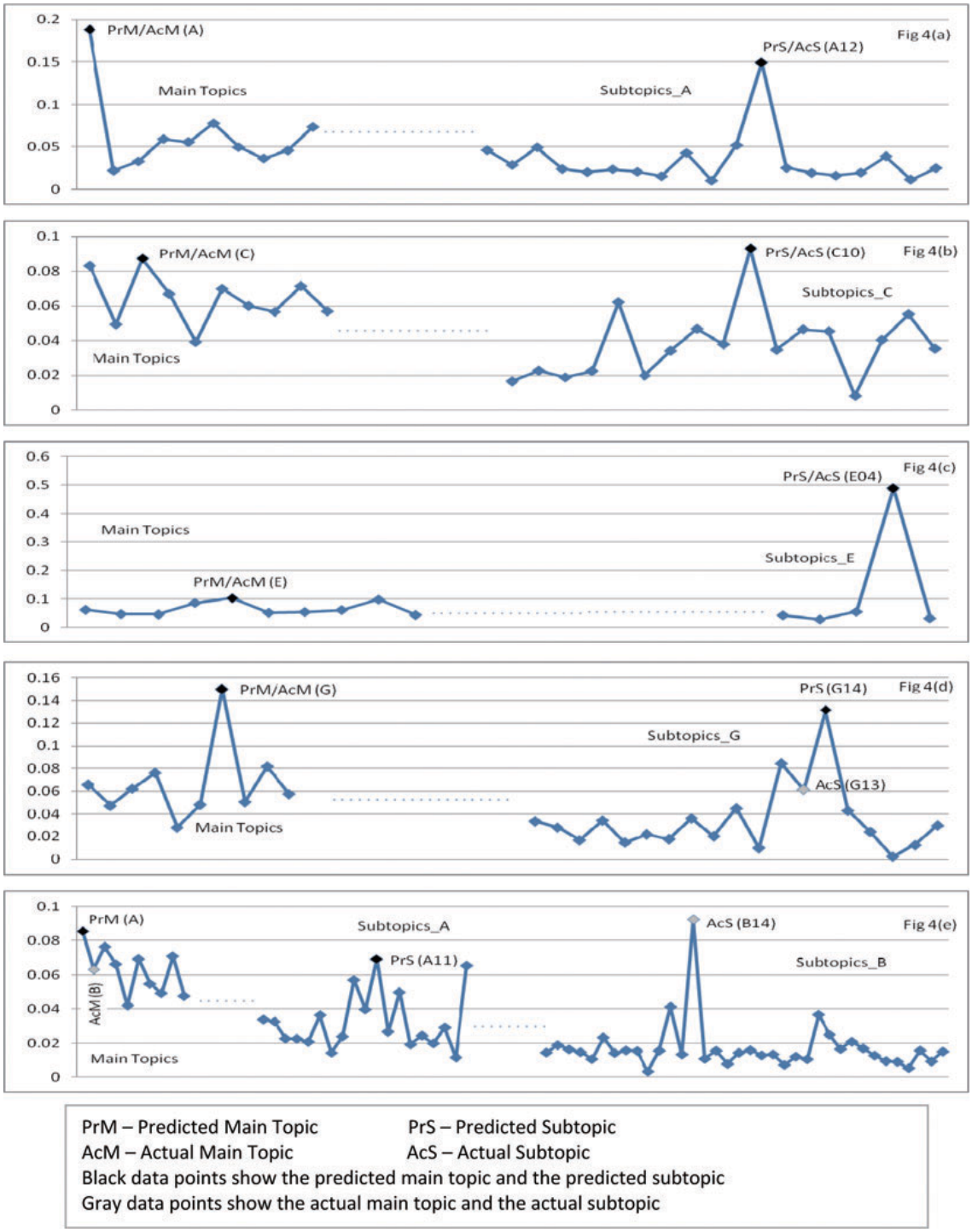

Fig. 4. Classification decisions by a hybrid parallel classifier for some LSHTC input vectors.

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

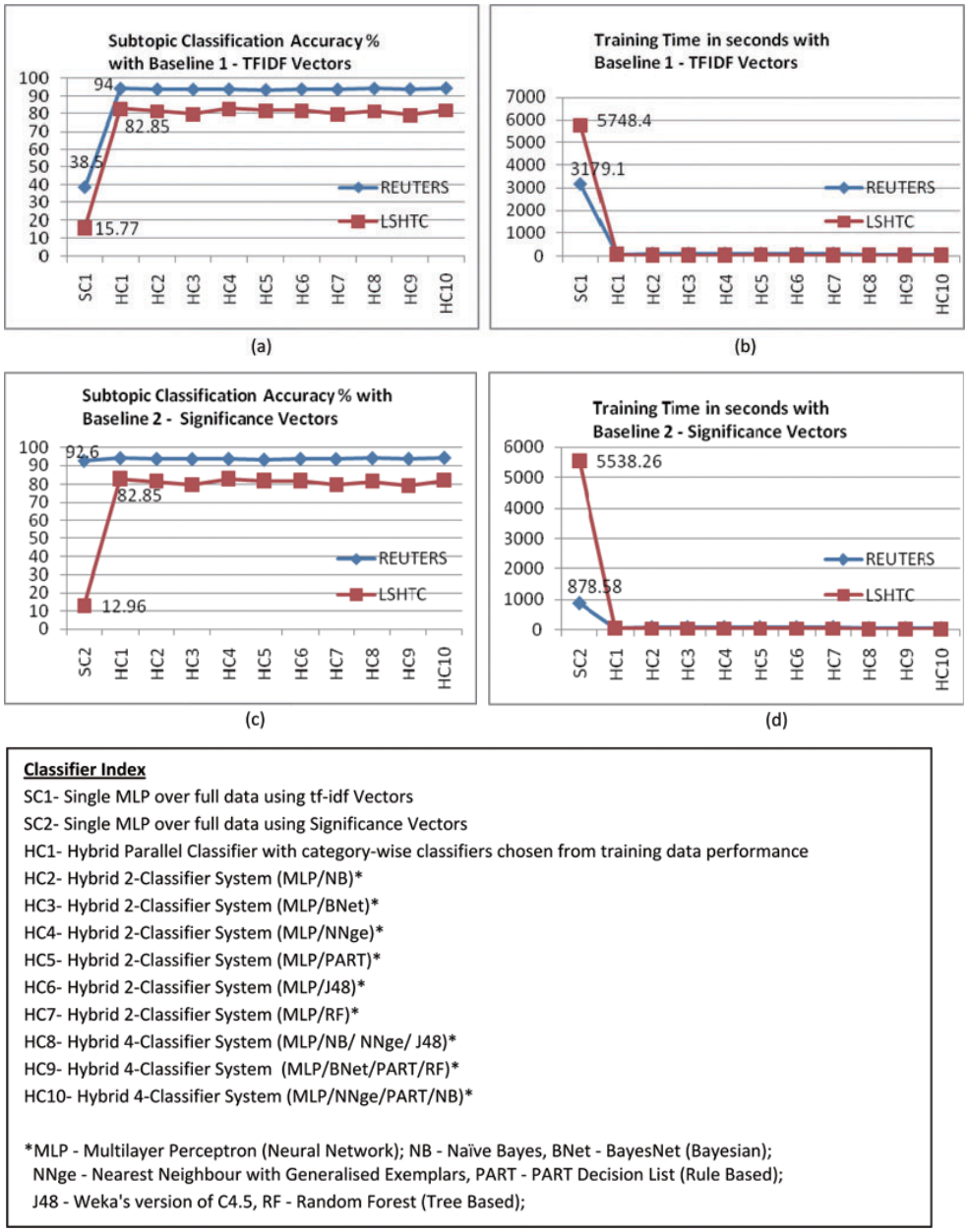

Fig. 5. Hybrid parallel classifiers performance metrics with baselines.

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

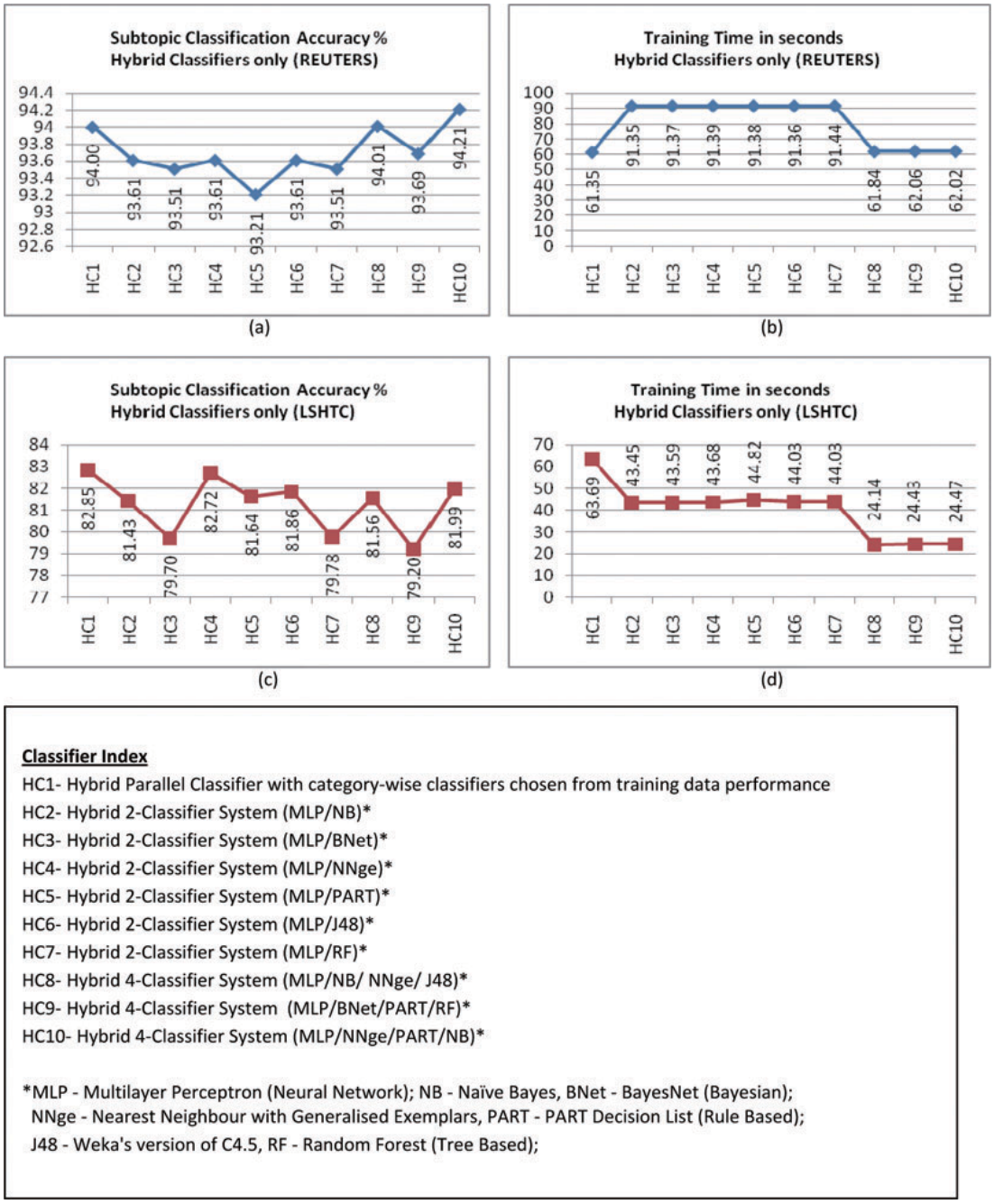

Fig. 6. Hybrid parallel classifiers only - Performance metrics.

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

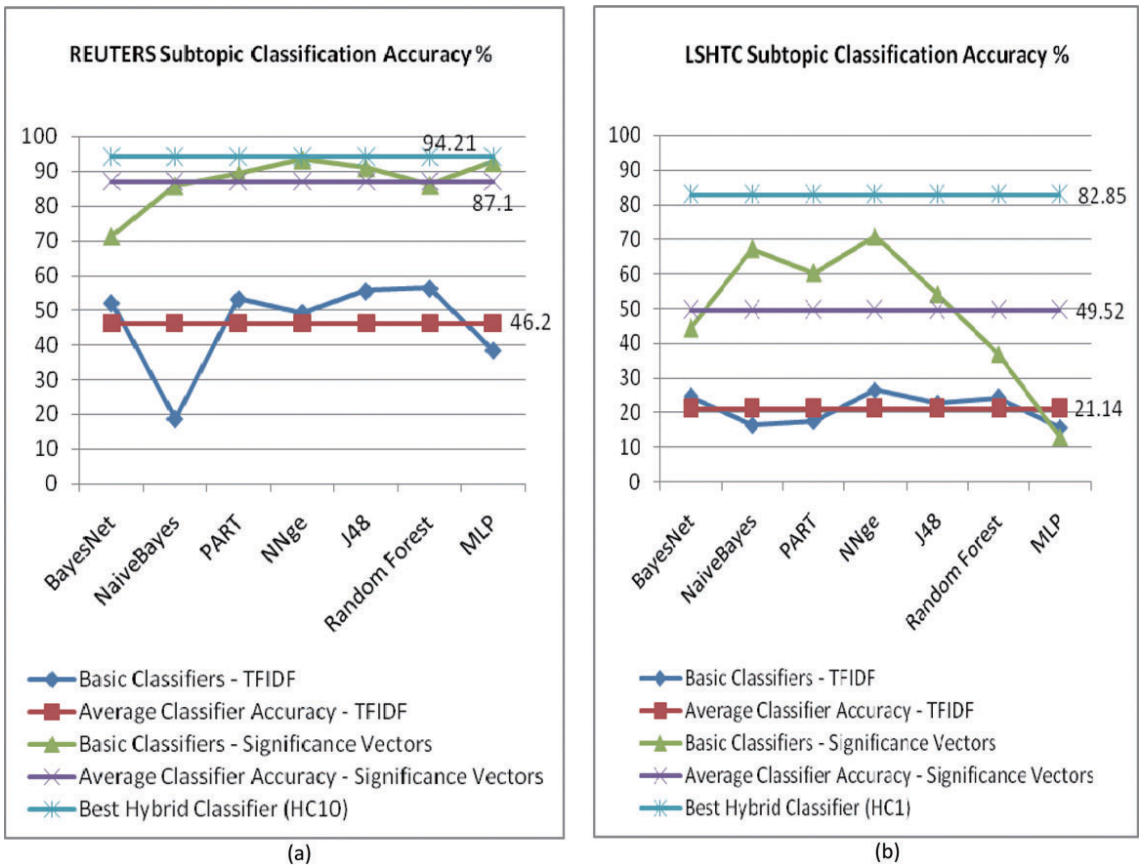

Fig. 7. Comparison of hybrid classifier performance with basic classifiers on full data.

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

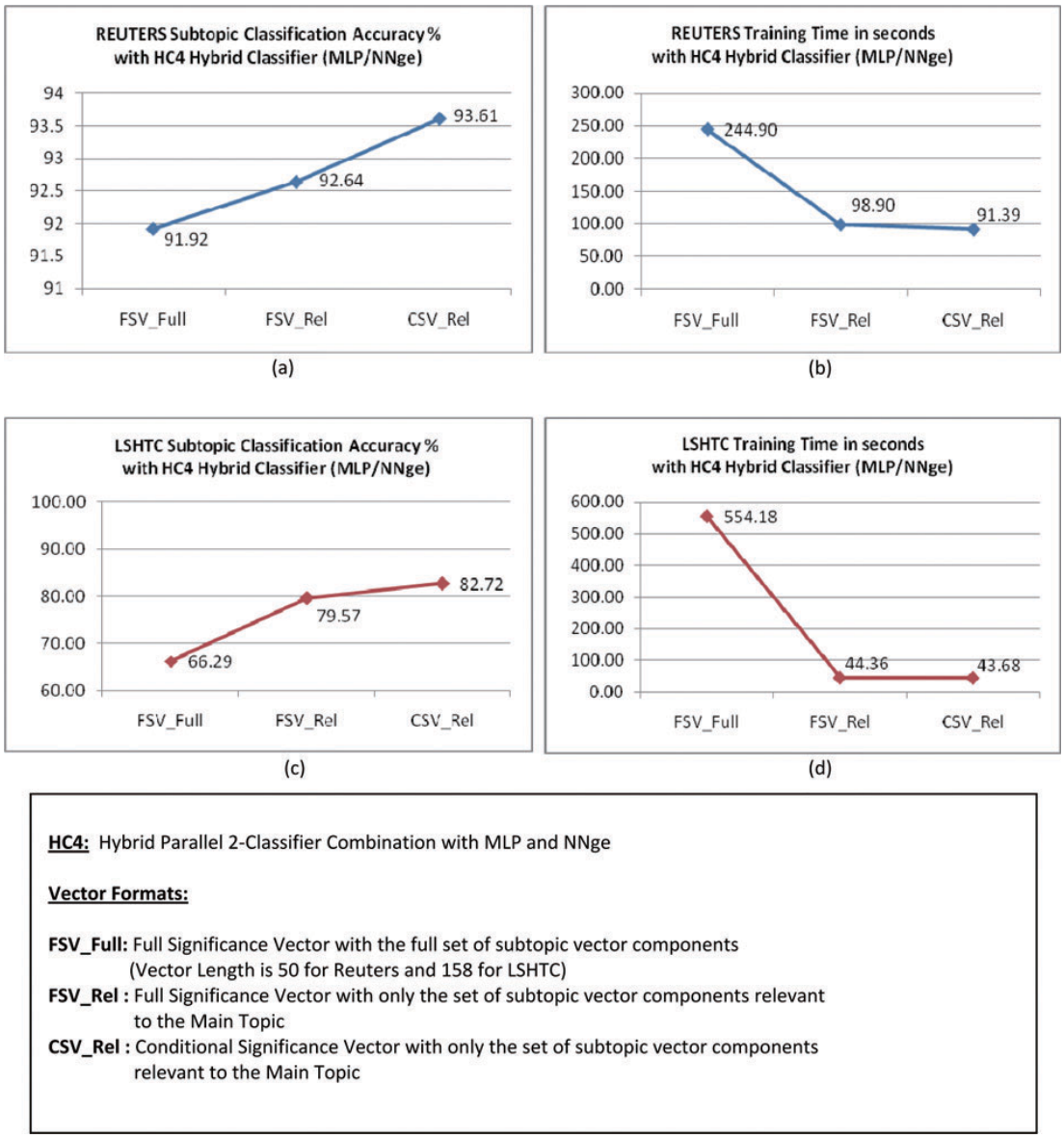

Fig. 8. Comparison of hybrid classifier (HC4) performance with different vector formats.

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).