2017 16th IEEE International Conference on Machine Learning and Applications

Text Classification using Hierarchical Sparse Representation Classifiers

Neeraj Sharma*, A. D. Dileep* and Veena Thenkanidiyoor†

*School of Computing and EE, Indian Institute of Technology Mandi, Mandi, H.P., India
†Department of CSE, National Institute of Technology Goa, Ponda, Goa, India

***Abstract*—In this paper, we propose to use sparse representation classifier (SRC) for text classification. The sparse representation of an example is obtained by using an overcomplete dictionary made up of term frequency (TF) vectors corresponding to all the training documents. We propose to seed the dictionary using principal components of TF vector representation corresponding to training text documents. In this work, we also propose 2-level hierarchical SRC (HSRC) by exploiting the similarity among the classes. We propose to use weighted decomposition principal component analysis (WDPCA) in the second level of HSRC to seed the dictionary to discriminate the similar classes. The effectiveness of the proposed approach to build HSRC for text classification is demonstrated on 20 Newsgroup Corpus.**

I. Introduction

The proliferation of Internet has led to massive amount of information in digital media. The textual content contributes to a large portion of it in addition to multimedia content. For a meaningful usage of such an ocean of information, an efficient mechanism for accessing such a huge repository is essential. Text classification is one important scheme in managing the huge repository. Text classification involves assigning a text document to one of the predefined class or a topic [1]. Usage of a wide variety of classifiers are explored for text classification and every such attempt aims to improve the performance of the classifier used. For example the simple naive Bayes classifier was found to perform well for text classification in [2]. Further, the text classification performance was found to be improved using support vector machines [3]. An important issue in using classifiers such as Bayes classifier, neural networks, nearest neighbor methods is the number of features used for representing a text document. To address this issue, many feature selection methods are considered for text classification [4], [5].

The focus of this work is to design an effective classifier for text classification. We use term-frequency (TF) representation for the text documents. In this paper, we explore a sparse representation classifier (SRC) for text classification. The sparse representation of data is a popular technique in signal processing. The sparse representation classifiers (SRCs) are extensively used in different image and speech processing tasks such as face recognition [6], [7], image classification [8], phonetic classification [9] and speaker verification [10]. To the best of our knowledge, except [1], no other serious attempt to use sparse representation for text classification. To obtain the sparse representation of data, a dictionary plays an important role. A dictionary $\mathbf{D}$ is a $d \times N_t$ matrix, where each column of the matrix is a d-dimensional training example and N_t is the number of training examples from all the classes. It is required for $\mathbf{D}$ to be an overcomplete dictionary such that the number of examples N_t is much larger than the dimension of each example ($d << N_t$). In signal processing domain, principal components of an example [7], [11] and discrete cosine transform of an example [11] are popularly used for seeding the dictionary. In [1], an overcomplete dictionary $\mathbf{D}$ is constructed using individual training documents with reduced vocabulary. In this work, we propose to use principle components of individual training documents to construct $\mathbf{D}$. Some of the document classes are highly confusable and it is observed that most of the examples are misclassified among themselves. To improve the classification performance, we exploit the similarity among the group of classes and propose to build 2-level hierarchical SRC (HSRC). First, we build SRC to classify a document into one of the groups (abstract classes) and this acts as SRC at the first level. Next, we build one SRC per abstract class to classify a document to their respective class within an abstract class. This acts as SRC at the second level. In order to better discriminate the classes within a group, we propose to use weighted decomposition principal component analysis (WDPCA) technique to obtain the middle principal components for constructing overcomplete dictionary in the SRCs at the second level of HSRC. We compare the performance of HSRC built using both PCA and WDPCA techniques to construct dictionaries. The effectiveness of all the techniques proposed in this paper are demonstrated for text classification on 20 Newsgroup Corpus [2].

The paper makes the following contributions towards exploring SRC for text classification. First, we explore seeding the dictionary using the principal components of all the training documents. This is in contrast to [1], where all the training documents with reduced vocabulary are used to seed dictionary. Building an HSRC and exploring it for text classification is the second contribution. Our third contribution is in exploring WDPCA technique to highlight the middle principal components to construct the dictionary, that emphasizes discrimination among confusing classes.

The rest of the paper is organized as follows. In Section II, sparse representation and classification based on sparse representation are presented. Hierarchical sparse representation classifier for document classification is presented in Section III. In Section IV, the studies on text classification is presented. The conclusions are presented in Section V.

0-7695-6321-X/17/31.00 ©2017 IEEE
DOI 10.1109/ICMLA.2017.00-18

II. Sparse representation classification

In this section we discuss about the sparse representation and classification using sparse representation.

A. Sparse representation

Given a feature vector $\mathbf{x} \in \mathbb{R}^d$ and a dictionary $\mathbf{D} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{N_t}]^\top \in \mathbb{R}^{d \times N_t}$ of N_t basis vectors, sparse representation of $\mathbf{x}$ aims to represent it as the linear combination of basis vectors as $\mathbf{x} = \beta_1 \mathbf{u}_1 + \beta_2 \mathbf{u}_2 + \cdots + \beta_{N_t} \mathbf{u}_{N_t}$. Here $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots \beta_{N_t}]^\top$ is the coefficient vector where β_n is the coefficient associated with the basis vector $\mathbf{u}_n$. The sparseness in representation is achieved by ensuring only a small fraction of elements in $\boldsymbol{\beta}$ to be non-zero. The problem of obtaining sparse representation, $\boldsymbol{\beta}$ of $\mathbf{x}$ can be formulated as

$$\min_{\boldsymbol{\beta}} \|\boldsymbol{\beta}\|_0 \text{ such that } \mathbf{x} = \mathbf{D}\boldsymbol{\beta} \tag{1}$$

Here $\|\boldsymbol{\beta}\|_0$ is the l_0-norm, which counts the number of nonzero entries in $\boldsymbol{\beta}$. However, the minimization of l_0-norm is an NP hard problem [12]. Recent developments in sparse representation [13] indicate that if the solution $\boldsymbol{\beta}$ is sparse enough, then l_0-norm in (1) can be replaced with an l_1-norm as

$$\min_{\boldsymbol{\beta}} \|\boldsymbol{\beta}\|_1 \text{ such that } \mathbf{x} = \mathbf{D}\boldsymbol{\beta} \tag{2}$$

The necessary condition for working with sparse representation based methods is that the dictionary $\mathbf{D}$ should be overcomplete, i.e., $d << N_t$.

B. Classification based on sparse representation

In the classification problem, training examples of all the classes act as basis vectors in $\mathbf{D}$. Now the dictionary matrix can be seen as $\mathbf{D} = [D_1, D_2, \ldots, D_M]^\top$, where M is the number of classes. Here, $D_m = [\mathbf{x}_{m1}, \mathbf{x}_{m2}, \ldots, \mathbf{x}_{mN_m}]^\top$ where $\mathbf{x}_{mn}$, for $n = 1, 2, \ldots, N_m$, are the training examples of the mth class. A test example belonging to mth class can be seen as the linear combination of training examples of mth class. For this example, the coefficients (β) associated with the training examples of mth class is non-zero and all the remaining coefficients are zero. The sparse vector $\boldsymbol{\beta}$, for the example is denoted as $\boldsymbol{\beta} = [0, \ldots, 0, \beta_{m1}, \beta_{m2}, \ldots, \beta_{mN_m}, 0, \ldots, 0]^\top$. Ideally the optimal $\boldsymbol{\beta}$ for a test example should be sparse and nonzero for the coefficients associated with the training examples belonging to the class of test example.

The main issue in considering sparse representation based methods for text classification is in having overcomplete dictionary matrix $\mathbf{D}$. The dimension of TF vector representation considered for each text document depends on the vocabulary size. Typically the size of vocabulary is larger than the number of training examples in a data set. When the TF vector representation of training examples are used as basis vector, the resulting $\mathbf{D}$ matrix becomes under complete, i.e., $d >> N_t$. This violates the necessary condition that $\mathbf{D}$ must be overcomplete. To address this issue, a reduced vocabulary ($d << N_t$) is considered in [1]. In this work, we are not reducing the vocabulary size. Instead, we propose to explore the principle component representation of TF vectors corresponding to documents to seed the dictionary. This approach is popular in sparse representation based face recognition [6]. In this work, we consider the leading principal components to ensure that the number of principal components to be less than the number of training documents. Dictionary derived using principle component representation of TF vectors corresponding to documents is given as

$$\hat{\mathbf{D}} = \boldsymbol{\Psi}^\top \mathbf{D} \tag{3}$$

where $\hat{\mathbf{D}}$ is the derived dictionary of the size $\hat{d} \times N_t$ and $\boldsymbol{\Psi}$ is the matrix consisting of $\hat{d}$ number of Eigen vectors corresponding to leading $\hat{d}$ number of Eigen values. These Eigen values and Eigen vectors are obtained from Eigen analysis of the set of TF vectors corresponding to training documents. Here, $\hat{d}$ should be smaller than or at most N_t.

We consider maximum l_2 support as classification rule [1] to assign a class label to a test example. In practice, elements of $\boldsymbol{\beta}$ corresponding to other than the class of test example could also be non-zero. To consider this fact, we compute the l_2-norm of the elements of $\boldsymbol{\beta}$ corresponding to every class and choose the class with largest l_2-norm as class label to the test example [10]. Let $\boldsymbol{\delta}_m(\boldsymbol{\beta})$ be a vector whose entries are the β values for the class m. Then assignment of the class label to a test example is given as

$$\text{class label} = \max_m (\|\boldsymbol{\delta}_m(\boldsymbol{\beta})\|_2); \text{ for all } m = 1, 2, \ldots, M \tag{4}$$

where M corresponds to number of classes.

In the next section we present the proposed hierarchical sparse representation classifier for text classification.

III. Classification based on Hierarchical Sparse Representation Classifier

In this section we first present the motivation for the hierarchical classification and then present proposed hierarchical sparse representation classifier (HSRC) for text classification.

Many times, the performance of a classification system will be poor when there exists high interclass similarity. This effect is more serious when the number of classes is large. One of the solutions to handle this situation is performing hierarchical classification. The process of performing hierarchical classification involves building classifiers to model the abstract classes at the different levels of hierarchy. At the top level (or first level), the similar classes are grouped to form the first level of abstract classes. A classifier is built at this level to classify an example into an abstract class. In the next level, the classes in each group may be further divided into sub groups. This heirarachy leads to a tree like structure and one can go to any depth in the tree based on how similar are the classes in each group. At each level, a classifier is built to classify the example either into a subgroups or into an actual class. The main advantage of such a hierarchy is that highly confusable classes can be managed well. At the same time, since number of classes in each group is small, the complexity of the classifier built at that level will be reduced. Motivated by these factors, we propose to use hierarchical classifier for text classification. In principle any classifier can be used at each level. However, we propose to use SRC at each level to take advantage of the discriminative capability imparted by dictionary. In this work, we propose an approach to build 2-level HSRC for text classification. In principle, one can extend this approach to any number of levels.

An important issue in building an HSRC is to identify the groups of classes at every level. In this work we form the groups of classes at first level of HSRC based on the confusion among the classes observed while classifying the examples according to decision rule in (4). The classes that are highly confused are put together in a single abstract class. Let M be the number of classes. Let Mg be the number of groups formed for the first level of HSRC. An SRC is built to classify the documents into one of the Mg number of abstract classes. This is the SRC at the first level of the HSRC built from the dictionary constructed using the principal components of the TF vectors corresponding to the documents belonging to the abstract classes. In the second level, an SRC is built for each of the Mg abstract classes considering the dictionaries constructed using the principal components of the TF vectors corresponding to the documents belonging to the classes corresponding to respective abstract classes. It is clear that, the SRC at the first level of HSRC is built to model the highly discriminative classes. On the other hand, the SRCs at the second level of HSRC are built to model the similar classes. Section II describes that dictionary play an important role in building the SRC. It is shown in [7] that first few principal components give average information of the classes and last few principal components contain least significant information of the class. Hence, the dictionary derived using middle principal components help in emphasizing subtle discriminative information of similar classes [7]. We propose to exploit this information in building the SRCs at the second-level of HSRC. We propose to use the principal components obtained using the Eigen vectors corresponding to the middle (not leading and not trailing) Eigen values in constructing dictionaries in the second-level SRCs to emphasize the discrimination among the confusing classes. However, it is difficult to come up with the best choice of start and end of middle principal components. In addition, this will result in decreased discrimination among distinct classes. This issue is addressed by modifying the dictionary as:

$$\hat{\mathbf{D}} = \mathbf{W}\mathbf{\Psi}^{\top}\mathbf{D} \tag{5}$$

where $\mathbf{W}$ is a diagonal weight matrix with $\left\{\frac{1}{\sqrt{\lambda_1}}, \frac{1}{\sqrt{\lambda_2}}, ..., \frac{1}{\sqrt{\lambda_d}}\right\}$ as diagonal elements. Here $\lambda_1, \lambda_2, ..., \lambda_d$ are the Eigen values corresponding to the Eigen vectors and $\mathbf{\Psi}$ is the matrix consisting of Eigen vectors. This transformed dictionary obtained from (5) is denoted as weighted decomposition (WD) of principal components. Hence, this technique is called as weighted decomposition principal component analysis (WDPCA). The WD transformation will allow the scaling of each principal component with the corresponding Eigen values and hence results in de-emphasizing the most significant principal components (corresponding to the largest Eigen values) and emphasizing the middle and last principal components. However, the last principal components does not contain any discriminative information and should be discarded. This can be done by removing the last principal components with the help of a thresholding operator.

In the next section we present our studies on text classification to evaluate the techniques introduced this paper.

IV. Experimental studies on text classification

In this section we first present the details of the data set used in the studies and features considered to represent documents. Next we present and discuss the results of the studies on text classification.

The 20 Newsgroup corpus [2] is used for evaluating the techniques introduced in this paper for text categorization. This corpus consists of 18,774 text documents divided into 20 different newsgroup classes. Among them 60% of the documents (i.e. 11,269) are used for training the models and remaining 40% (i.e. 7,505) of the documents are used for test. The text document categorization accuracy presented is the classification accuracy obtained for the test examples. In this study, we considered term-frequency (TF) as a feature. We have considered 53,975 terms (or words) from all the training documents as the vocabulary. The frequency of occurrence of each of the vocabulary term in a text document is computed. Every document is represented as a 53,975-dimensional TF vector.

A. Studies using SRC and SVM-based classifier

First, we present the studies on text classification using sparse representation classifier (SRC). SRC requires the construction of dictionary matrix $\mathbf{D}$ by considering the training examples represented as TF vector at every column. For the 20 Newsgroup corpus, the number of documents ($N_t = 11,269$) is much smaller than the dimension of the TF vector representation for documents ($d = 53,975$). For the success of SRC, we need $d << N_t$. To comply with this requirement, we applied principal component analysis (PCA) on the document vectors. This involves projecting a document vector along the Eigen vectors corresponding to the leading Eigen values. Then we used principal component representation in $\hat{\mathbf{D}}$. The performance of SRC is analyzed by building $\hat{\mathbf{D}}$ with the number of principal components $\hat{d}$ taking the values from 1,000 to 11,000 with the increment of 1,000. The best classification accuracy of SRC using maximum l_2 support is observed to be **78.78**% for $\hat{d}$=6,000, which is comparable with that of the best accuracy (78.80%) reported in [1]. In [1], the vocabulary size is reduced to 11,000 and TF vector representation is obtained using this vocabulary.

Next, we present the studies on text classification using SVM classifier using Gaussian kernel (GK). In this study we considered TF vector representation with dimension $d = 53,975$ for each text document. We consider SVMTorch [14] tool to build the SVM-based classifiers. In this study, one-against-the-rest approach is considered for 20-class text document classification. The value of trade-off parameter, C in SVM is chosen empirically. The classification accuracies for the SVM-based classifier using GK are given in Table I for text classification. It is seen that the performance of the proposed SRC is marginally better than the GK based SVMs.

TABLE I. Classification accuracy (CA) (in %) of the SRC and SVM-based classifiers using Gaussian kernel (GK) for text classification. Here TF is term frequency.

Representation	Classification model		CA
TF	SVM using	GK	77.30
Principal component based	SVM using	GK	77.70
	SRC	-	78.78

The misclassification observed in 20 Newsgroup corpus can be attributed to the presence of highly confusing classes. It is

TABLE II. Confusion matrix (in %) obtained using SRC for the 20 classes.

Class Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	72	0	0	0	0	1	0	1	0	0	0	2	1	3	2	13	2	1	0	3
2	0	63	4	4	3	8	3	1	1	1	1	2	4	1	2	1	0	1	0	0
3	0	4	55	14	3	8	2	1	1	0	1	2	3	2	1	3	0	1	1	0
4	0	2	4	76	6	1	4	0	0	1	1	1	5	0	0	0	0	0	0	0
5	0	1	3	8	72	3	3	1	1	1	1	2	2	0	1	1	1	0	1	0
6	0	9	6	4	2	70	1	1	1	1	0	2	1	1	1	1	1	0	0	0
7	0	1	1	9	4	2	69	4	1	2	1	0	2	0	1	1	0	0	0	0
8	0	1	1	1	1	0	4	80	6	1	1	0	3	1	1	1	1	0	1	0
9	0	0	0	1	0	0	0	3	94	1	0	0	1	1	0	0	0	0	0	0
10	1	0	0	0	0	0	1	0	0	90	6	0	0	0	0	0	1	0	0	0
11	0	0	0	0	0	1	0	0	0	1	97	0	0	0	0	1	0	0	0	0
12	0	0	1	1	0	0	1	0	1	1	0	90	1	1	0	1	2	0	0	0
13	1	2	2	6	3	2	3	2	3	0	1	4	68	2	4	0	0	0	0	0
14	2	1	0	1	1	1	1	1	2	1	1	1	2	81	0	5	1	0	0	0
15	0	2	0	1	1	1	1	1	2	1	0	2	2	2	82	0	1	1	0	0
16	2	0	0	0	1	0	0	0	0	1	1	0	0	1	1	92	0	0	0	1
17	0	1	0	2	0	0	0	1	0	1	1	2	0	2	1	2	84	0	3	1
18	5	1	0	1	0	0	1	0	2	1	0	1	0	0	0	4	1	82	1	0
19	1	0	0	0	1	0	1	1	1	1	1	2	0	2	2	2	29	1	53	1
20	10	0	0	0	0	0	1	0	1	2	0	0	0	2	2	28	7	1	2	43

TABLE III. Combining of confusable classes to form abstract classes. Here the number attached to class name corresponds to its number.

Abstract-Class1	Abstract-Class2	Abstract-Class3	Abstract-Class4	Abstract-Class5
1. Atheism	2. ComputerGraphics	8. RecAutos	10. SportBaseball	15. SciSpace
14. SciMedi	3. CompOsMsWindowsMisc	9. RecMotorcycle	11. SportHockey	
16. ReligionChristian	4. CompSysIbmPcHardware			
17. PoliticsGuns	5. CompSysMacHardware			
18. PoliticsMidEast	6. CompWindowsX			
19. PoliticsMisc	7. MiscForsale			
20. ReligionMisc	12. SciCrypt			
	13. SciElectronics			

also observed that the confusing classes are also semantically related. We noticed that some of the text documents which should have been classified correctly are getting misclassified due to the similarity with other classes and in this way SRC's performance was getting affected. These semantically related classes can be grouped into a single abstract class and a hierarchical classifier built using SRC is expected to perform better. This motivated us to use SRC hierarchically. The confusion matrix for SRC is given in Table II. Based on the confusion matrix for SRC, we combined the classes which were getting highly confused with each other as shown in Table III. It is seen that Abstract-Class1 consist of 2 different type of classes, one related to religion and other are related to politics. In the similar fashion Abstract-Class2 consists of computer science related document classes, classes related to electronics and cryptography related documents. Abstract-Class3 consists of automobile related classes. Abstract-Class4 contains classes related to sports documents and Abstract-Class5 contains only one document class.

B. Studies using hierarchical SRC

An hierarchical SRC (HSRC) is built in two levels. In the first level, an SRC is built such that the number of classes equal to number of abstract classes. The same $\hat{\mathbf{D}}$ mentioned in Section IV-A is used as dictionary. Note that $\hat{\mathbf{D}}$ is constructed by using principal component representation of each document obtained by projecting TF vector along the Eigen vectors corresponding to the leading Eigen values. In the second level, an SRC is built for each of the abstract classes separately. Now $\hat{\mathbf{D}}_i$ for ith abstract class is obtained using WDPCA technique as explained in Section III. Here, $\hat{\mathbf{D}}_i$ is constructed by using principal component representation of each document belonging to the classes in ith abstract class obtained by projecting document vector along the Eigen vectors corresponding to the middle Eigen values. Now each test example is first classified into one of the 5 abstract classes. Then it will be classified into a document class within that abstract class. The classification accuracy for HSRC is given in Table IV for text classification. It is seen that **83.30%** of the documents are correctly classified using HSRC as opposed to the 78.78% using SRC. The reason for this 6% increase in the accuracy is mainly due to the use of WDPCA in constructing $\hat{\mathbf{D}}_i$ for SRC of ith abstract class. The fact is that the WDPCA highlights the middle principal components that are responsible for discrimination among similar classes.

TABLE IV. Classification accuracy (CA) (in %) of the proposed HSRC and SRC for text classification.

Representation	Classification model	CA
Leading PCA	SRC	78.78
WDPCA	HSRC	**83.30**

C. Empirical analysis of effectiveness of WDPCA in constructing SRCs at second level

In this section, we present the effectiveness of WDPCA in constructing dictionary for the group of classes that are similar. To show this we compare the performance of SRC built for each abstract class using WDPCA-based dictionary with that of the SRC built for each abstract class using PCA-based dictionary. Note that, here SRC for an abstract class is built using the training examples of all the similar classes belonging to it. Test accuracy is computed using only the test examples belonging to the classes in that abstract class. The classification performance of each class in every abstract class using SRC with PCA and WDPCA based dictionary is compared in the

Tables V - VIII. The results for Abstract-Class5 is not shown, as there is only one document class in that abstract class.

TABLE V. Classification accuracy (CA) (in %) of each class in Abstract-Class1 and their comparison when the SRC is built using PCA and WDPCA based dictionary.

Class Name	PCA	WDPCA
1. Atheism	73.00	84.00
14. SciMedi	78.00	89.00
16. ReligionChristian	81.00	90.00
17. PoliticsGuns	72.00	85.00
18. PoliticsMidEast	75.00	82.00
19. PoliticsMisc	71.00	83.00
20. ReligionMisc	79.00	86.00

TABLE VI. Classification accuracy (CA) (in %) of each class in Abstract-Class2 and their comparison when the SRC is built using PCA and WDPCA based dictionary.

Class Name	PCA	WDPCA
2. ComputerGraphics	55.00	67.00
3. CompOsMsWindowsMisc	61.00	76.00
4. CompSysIbmPcHardware	69.00	84.00
5. CompSysMacHardware	55.00	78.00
6. CompWindowsX	61.00	72.00
7. MiscForsale	69.00	79.00
12. SciCrypt	84.00	92.00
13. SciElectronics	73.00	85.00

TABLE VII. Classification accuracy (CA) (in %) of each class in Abstract-Class3 and their comparison when the SRC is built using PCA and WDPCA based dictionary.

Class Name	PCA	WDPCA
8. RecAutos	81.00	93.00
9. RecMotorcycle	82.00	85.00

TABLE VIII. Classification accuracy (CA) (in %) of each class in Abstract-Class4 and their comparison when the SRC is built using PCA and WDPCA based dictionary.

Class Name	PCA	WDPCA
10. SportBaseball	93.00	96.00
11. SportHockey	80.00	93.00

It is seen that the SRC built using WDPCA-based dictionary performs significantly better than that of the SRC built using PCA-based dictionary. This is the empirical evidence of the significance of middle principal components in discriminating the similar classes as explained in [7]. The main drawback of the WDPCA is that it works better only if grouping technique is good enough and only similar type of classes are grouped together. If grouping technique is not good, WDPCA can affect the performance of SRC.

V. Conclusions

Approaches to text casification using sparse representation classifiers (SRC) is explored in this paper. To obtain sparse representation, an overcomplete dictionary (number of columns being much larger than the number of rows) made up of training documents is used. In text classification, typically dictionaries turn out to be under complete. To overcome this issue, we built an overcomplete dictionary using the principal components based representation corresponding to the training documents. Hierarchical SRC (HSRC) is also proposed in this work, in which we proposed use WDPCA technique to see the dictionary matrix. HSRC works better than SRC because of confusing documents of similar type of classes affects the performance of SRC but after grouping such classes together that problem gets resolved. WDPCA works well in the field of document classification but WDPCA technique needs a very good grouping technique as WDPCA is used to discriminate between similar type of documents. The effectiveness of the proposed techniques is demonstrated using 20 Newsgroup Corpus. These approaches need to be explored with other datasets also. Possibly using the semantic information may also improve the performance of text classification.

References

[1] Tara N Sainath, Sameer Maskey, Dimitri Kanevsky, Bhuvana Ramabhadran, David Nahamoo, and Julia Hirschberg, "Sparse representations for text categorization.," in *Proceedings of INTERSPEECH*, 2010, vol. 10, pp. 2266–2269.

[2] Andrew McCallum and Kamal Nigam, "A comparison of event models for naive Bayes text classification," in *Proceedings of AAAI-98 workshop on learning for text categorization*, 1998, vol. 752, pp. 41–48.

[3] Thorsten Joachims, "Text categorization with suport vector machines: Learning with many relevant features," in *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*. ACM, 1998, pp. 137–148.

[4] Anirban Dasgupta, Petros Drineas, Boulos Harb, Vanja Josifovski, and Michael W. Mahoney, "Feature selection methods for text classification," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'07)*, 2007, pp. 230–239.

[5] George Forman, "An extensive empirical study of feature selection metrics for text classification," *Journal of Machine Learning Research*, vol. 3, pp. 1289–1305, 2003.

[6] John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, February 2009.

[7] T. Shejin and Anil Kumar Sao, "Significance of dictionary for sparse coding based face recognition," in *Proceedings of 11th International Conference of the Biometrics Special Interest Group*, Darmstadt, Germany, September 2012, pp. 1–6.

[8] Xiao-Tong Yuan and Shuicheng. Yan, "Visual classification with multitask joint sparse representation," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4349–4360, October 2012.

[9] Tara N Sainath, Avishy Carmi, Dimitri Kanevsky, and Bhuvana Ramabhadran, "Bayesian compressive sensing for phonetic classification," in *Proceedings of IEEE International Conference onAcoustics Speech and Signal Processing (ICASSP 2010)*, 2010, pp. 4370–4373.

[10] Jia Min Karen Kua, Julien Epps, and Eliathamby Ambikairajah, "i-Vector with sparse representation classification for speaker verification," *Speech Communication*, vol. 55, no. 5, pp. 707 – 720, 2013.

[11] M. Elad, *Sparse and Redundant Representations - From Theory to Applications in Signal and Image Processing*, Springer, 2010.

[12] Edoardo Amaldi and Viggo Kann, "On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems," *Theoretical Computer Science*, vol. 209, no. 12, pp. 237 – 260, 1998.

[13] David L Donoho, "For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution," *Communications on Pure and Applied Mathematics*, vol. 59, no. 6, pp. 797–829, 2006.

[14] R. Collobert and S. Bengio, "SVMTorch: Support vector machines for large-scale regression problems," *Journal of Machine Learning Research*, pp. 143–160, 2001.