# *SR-NBS: A fast Sparse Representation based N-best class Selector for robust phoneme classification*

*Armin Saeb*[a], *Farbod Razzazi*[a], *Massoud Babaie-Zadeh*[b]

[a]Department of Electrical and Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran.

[b]Electrical Engineering Department, Sharif University of Technology, Tehran, Iran.

ABSTRACT

Although exemplar based approaches have shown good accuracy in classification problems, some limitations are observed in the accuracy of exemplar based automatic speech recognition (ASR) applications. The main limitation of these algorithms is their high computational complexity which makes them difficult to extend to ASR applications. In this paper, an N-best class selector is introduced based on sparse representation (SR) and a tree search strategy. In this approach, the classification is fulfilled in three steps. At first, the set of similar training samples for the specific test sample is selected by k-dimensional (KD) tree search algorithm. Then, an SR based N-best class selector is used to limit the classification among certain classes. This makes the classifier adapt to each test sample and reduces the empirical risk. Finally, a well known low error rate classifier is trained by the selected exemplar samples and the trained classifier is employed to classify among the candidate classes. The algorithm is applied to phoneme classification and it is compared with some well-known phoneme classifiers according to accuracy and complexity issues. By this approach, we obtain competitive classification rate with promising computational complexity in comparison with the state of the art phoneme classifiers in clean and well known acoustic noisy environments which causes this approach become a suitable candidate for ASR applications.

**Keywords:** N-best Class Selector, Sparse Representation, Phoneme Classification, KD-Tree, Support Vector Machines

# 1. INTRODUCTION

Phoneme classification is the labeling procedure of isolated speech segments with the most likely phonetic labels. Phoneme classification plays a key role in most of automatic speech recognition (ASR) algorithms. The final performance of an ASR system basically depends on the phoneme classifier accuracy (Rabiner, 1989; Juang and Rabiner, 2005; Keshet et al., 2007).

In traditional phoneme classifiers, it is assumed that the training ensembles can represent the behavior of the test unseen data. The model parameters are determined by a set of training utterances and the resulted model is then employed to classify the test utterances. As a result, model parameters are not adapted to test examples and the empirical risk of classification can be high. In addition, in noisy environments, the model is usually trained by both noisy and noiseless utterances to make the system robust against different signal to noise ratios. Indeed, it makes the model parameters be matched to the average of noisy samples, while it is better to adapt the system to the specific noisy situation as in the test input. The mismatch between noise specifications of training and test utterances causes the accuracy of the phoneme classifier decrease in noisy environments in comparison to the situation in which the noise specifications of the training and test utterances are completely matched (Amrouche et al., 2010). Although most practical speech models have been developed based on statistical modeling of speech (e.g. Gaussian Mixture Models (GMM), Hidden Markov Models (HMM)), exemplar based classifiers have shown their superiority in non-speech classification applications (Joachims, 2002; Tzicas et al., 2006; Wang 2010) and clustering (Ping et al., 2010). Exemplar-based classifiers like K-Nearest Neighbors (KNN), Support Vector Machines (SVM) (Vapnik, 1998) and Relevance Vector Machines (RVM) (Tipping, 2001) have tried to decrease the empirical risk of classification by adapting the model parameters to the details of the training samples, not to the average behavior of them. However, due to large number and vast variety of training samples in speech recognition, most of the training samples try to incorporate in the classification procedure, which makes the model impractical (e.g. the large number of support vectors (SVs) in an SVM phoneme classifier (Li, 2008) or the required exhaustive search in an KNN phoneme classifier). In addition, in conventional exemplar-based methods, the model is fully estimated by the training samples and the test utterance is not incorporated to adapt the model parameters. Therefore, their classification accuracy is limited in ASR real world applications. Although some online learning algorithms like Passive Aggressive (PA) algorithms (Crammer et al., 2006) are potentially capable to add the test utterance for complementary training; but they need some suitable confidence criteria that it has not been presented yet; up to our best knowledge.

Sparse signal representation (SR), a technique to represent a signal by a small number of basic signals (atoms) (Chen et al., 1999; Elad, 2010), has been shown to be successful in many signal processing application (e.g. face recognition (Wright et al., 2009), phoneme classification (Sainath et al., 2010) and multi-party speech recovery (Asaei et al., 2011)). In sparse representation, an $n \times 1$ signal vector $\mathbf{y}$ is represented by a linear combination of $m$ basic $n \times 1$ signal vectors (atoms) $\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_m$, so that $\mathbf{y} = \mathbf{X} \,.\, \boldsymbol{\lambda}$ where $n < m$. As $n < m$ the solution is not unique. But as we want to use the minimum number of atoms (i.e. sparse solution), the coefficient vector $\boldsymbol{\lambda}$ would be unique in most of practical cases (Donoho, 2006).

Sparse representation seems to be an appropriate approach in classification problems where there is sufficiently a large number of training samples. Due to the existence of good standard transcribed corpora in speech processing, phoneme classification meets this condition well. It is because when there is a large number of training examples, the chance of similarity between the test sample and a sparse set of training examples will be increased and it would be more likely that the test sample can be stated as a linear combination of a sparse set of training samples. Sainath et al have used this approach for phoneme classification (Sainath et al., 2010) and have extended their method to large vocabulary continuous speech recognition (LVCSR) (Sainath et al., 2011a). They have used Approximate Bayesian Compressive Sensing (ABCS) algorithm as the sparse representation method that has been reported by IBM research group (Carmi et al., 2009). Gemmeke et al have employed SR for noisy speech modeling as a linear combination of speech and noise samples. They have used this strategy for noise robust digit recognition (Gemmeke and Virtanen, 2010). They also have used SR as a missing data technique (MDT) to estimate the clean speech features from noisy speech signal (Gemmeke et al., 2010). LASSO algorithm (Tibshirani, 1996) has been the sparse representation method in their study. Both of the above algorithms are based on $l^1$ or $l^2$ norm minimization which are computationally too complex and time consuming. Therefore, as stated in (Sainath et al., 2011a), implementing an exemplar based method in LVCSR applications has been reported as a computationally hard approach.

In this paper, we use a new classifier architecture using a fast $l^0$ norm SR algorithm for phoneme classification. This SR algorithm, which is called smoothed $l^0$ (SL0), has been firstly introduced by Mohimani et al (Mohimani et al., 2009). Although previous approaches in using SR in phoneme classification have used it only as the classification engine, we propose that SL0 approach may be regarded as a training set selector for a classic pattern recognition system. This tunes the model to the test sample with limited computational complexity. By using this approach, we present a two stage classifier named as sparse representation based N-best class selector (SR-NBS). In this classifier, the N-best classes are chosen by using a searching strategy and the SL0 algorithm. Subsequently, a low error rate classifier (e.g. SVM) is trained by the selected samples and the classifier is employed to classify the test utterance among the candidate classes. Simulations show that this method results in a noticeable phoneme classification rate in a fair computational complexity, outperforming previously proposed exemplar based classifiers. Basic ideas of this paper and preliminary results of the simulations were presented in (Saeb and Razzazi, 2012).

The rest of the paper is organized as follows. In section 2, the general framework for sparse classification is explained, making the framework compatible to phoneme classification problem. In this section, SL0 algorithm and some of its properties will be explained too. In section 3, we explain the proposed algorithm. The motivation, mathematical formulation and computational complexity analysis of the algorithm are presented in this section. The experimental results of evaluation of the idea on a phoneme classification benchmark and comparison with other well known classifiers in noiseless and noisy environments are presented in section 4. Finally, section 5 concludes the paper and discusses some future works.

## 2. SR APPROACH FOR PHONEME CLASSIFICATION

### 2.1 General Framework

In traditional exemplar based classification frameworks, the classification procedure is indeed the approach to employ the training set to find out the most similar training samples given a test sample. For example, as depicted in Figure 1, the received test sample is classified as class 2 in nearest neighborhood approach, because the most training samples in its neighborhood belong to this class.

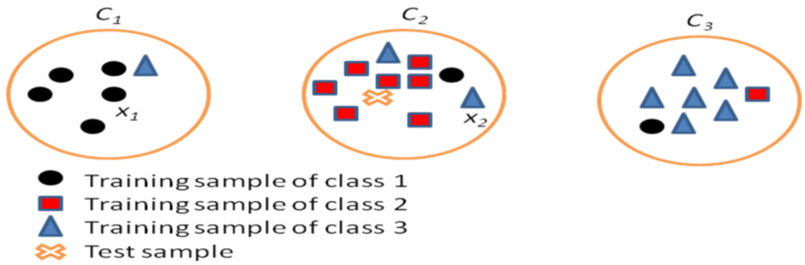



**Fig. 1. An example of classification: the test sample is classified as class 2**

The main motivation for sparse classification is the fact that if there are sufficient training samples from the class that the test sample belongs to, the test sample can be represented as a linear combination of the nearest training samples (Wright et al., 2009). However, the optimization problem is underdetermined and there are many combinations to represent the sample. But the sparse combination of the training samples that has ideally a few nonzero elements is the desired combination.

Suppose that $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ are the training $n \times 1$ utterance vectors where n is the number of extracted features and m>>n (the hypothesis that is usually correct in speech recognition and other pattern recognition problems with sufficient training data). Due to the large number of training samples, it is expected that the test utterance vector $\mathbf{y}$ can be determined by a linear combination of a few neighboring training utterance vectors of the same class of the test vector. Therefore, classification may be performed by the following equation based on SR model:

$$j^* = Arg\max_j f(\lambda_j) \tag{1}$$

where $f(.)$ is a function which is determined by the classification rule and $\boldsymbol{\lambda}^* = [\lambda_1, \lambda_2, \ldots, \lambda_m]$ is a sparse vector resulted from SR optimization:

$$\boldsymbol{\lambda}^* = Arg\min \|\boldsymbol{\lambda}\|_0 \quad subject\ to \quad \mathbf{y} = \mathbf{X}.\boldsymbol{\lambda} \tag{2}$$

where $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}]$.

Unfortunately, there are some challenges to apply (2) to phoneme classification directly. At first, as $m$ and $n$ increase, solving (2) is an NP-hard problem. Therefore, some researchers have examined other approaches and focused on convex relaxations of $l^0$ norm like $l^1$ norm (e. g. LASSO algorithm (Tibshirani, 1996)) or non convex $l^p$ norm (0<p<1) (e. g. FOCUSS algorithm (Gorodnitsky and Rao, 1997)). In some algorithms such as Approximate Elastic Net (EN) (Zou and Cole, 2005) or ABCS (Carmi et al., 2009), a combination of $l^1$ and $l^2$ norms have been used and better results have been obtained in comparison of $l^1$ norm (Sainath et al., 2011a). Although these algorithms are tractable, they are still slow, especially in LVCSR applications (Sainath et al., 2011a). Therefore, it seems that using an SR algorithm with low complexity and reasonable accuracy instead of a complex SR algorithm would be a good approach in employing SR in LVCSR applications. Recently, minimizing $l^0$ norm in (2) has been noticed by some researchers and they have tried to solve (2) directly, without substituting with $l^1$, $l^2$ or $l^p$ norms (e.g. (Ulfarsson and Solo, 2011; Ulfarsson and Solo, 2012; Seneviratne and Solo, 2012)). Among the approaches for $l^0$ norm minimization, SL0 (Mohimani et al., 2009) is one of the most successful approaches with low complexity and appropriate accuracy. In SL0, the $l^0$ norm term $||\boldsymbol{\lambda}||_0$ of (2) is substituted by a suitable continuous function of $\boldsymbol{\lambda}$. In the next subsection, this algorithm will be briefly reviewed.

The second challenge is the classification rule. Equation (2) only gives the coefficients of the sparse decomposition. It is expected that $\boldsymbol{\lambda}$ has nonzero elements only in locations that corresponds to the training vectors with the same label as the test vector $\mathbf{y}$. Therefore, if $\mathbf{y}$ and the training vectors $\mathbf{x}_k$ ($k$=1, 2, …, $m$) belong to the classes $c_y$ and $c_k$ respectively, we expect to have $\lambda_k \neq 0$ only if $c_y = c_k$. In addition, if there are enough training samples, we expect that the test sample corresponds to one of the training samples and therefore one of the $\lambda_k$'s is much greater than the others. However, in the classification problems that the classes are not separable (e.g. phoneme classification), this is not true and $\boldsymbol{\lambda}$ has many nonzero elements. Therefore, we need a suitable classification rule to determine the class of $\mathbf{y}$ from the coefficient vector of the sparse solution $\boldsymbol{\lambda}$. In (Wright et al., 2009) and (Sainath et al., 2011a), some of these decision rules are examined. If we define $m\times 1$ vector $\boldsymbol{\delta}_c(\boldsymbol{\lambda})$ as the vector that its elements are zero except for the elements in $\boldsymbol{\lambda}$ corresponding to class $c$, some examined decision rules are:

$$c^* = Arg\,max(\boldsymbol{\lambda}) \qquad \text{(Maximum Support)} \tag{3}$$

$$c^* = Argmax\|\boldsymbol{\delta}_{\mathbf{c}}(\boldsymbol{\lambda})\|_2 \qquad \text{(Maximum } l^2 \text{ Support)} \tag{4}$$

$$c^* = Argmin\|\mathbf{y} - \mathbf{X}.\boldsymbol{\delta}_{\mathbf{c}}(\boldsymbol{\lambda})\|_2 \qquad \text{(Minimum Class Residual Error)} \tag{5}$$

Although the class that minimizes the intra-class residual error in the selected sparse samples is selected in (5); it is not necessarily implies that this strategy minimizes the total residual error (Sainath et al., 2011a). In this paper, we have used (4) as the classification rule.

The third challenge is the selection of *m* and **X** in (2). We cannot use all training vectors in (2) for two reasons. First, in phoneme classification, the number of training vectors is usually very large (at least one hundred thousand vectors in small vocabulary experiments) and if the whole set of training vectors is employed in the sparse decomposition, solving (2) will be complex and intractable. Secondly, in sparse classification approach, it is better to use the training vectors near the test vector to decrease the errors that may happen from unsuitable sparse combinations. The reason is, as it is shown in Figure 2, the test sample can be exactly expressed as the linear combination of two far training samples $x_1$ and $x_2$ driven from wrong classes.

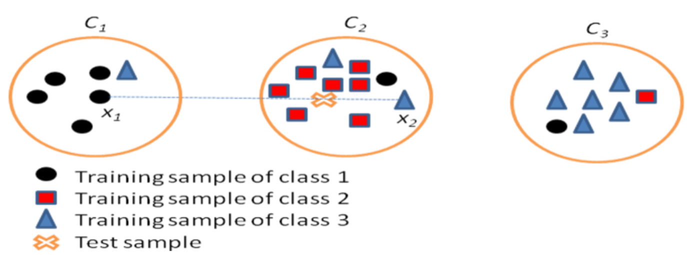



**Fig. 2. An example of sparse classification error: the test sample can be stated as linear combination of training samples $x_1$ and $x_2$**

Obviously, this combination is sparse, but not suitable. Although there are some methods that have been used to select appropriate exemplars (e. g. seeding **X** from nearest neighbors (Sainath et al., 2010) or using a Trigram Language Model (Soan et al., 2003)) , this problem is still open and new algorithms may improve the classifier's complexity and accuracy. In this paper, we have employed the KD-tree search strategy and have seeded **X** by the nearest neighbors of each test sample which have been chosen from the tree.

The block diagram of a general sparse classifier is shown in Figure 3. First, the matrix **X** is constructed by a search strategy. Then, SR algorithm determines the coefficient vector **λ**. Finally, the class of the test vector is estimated by one of the discussed rules.

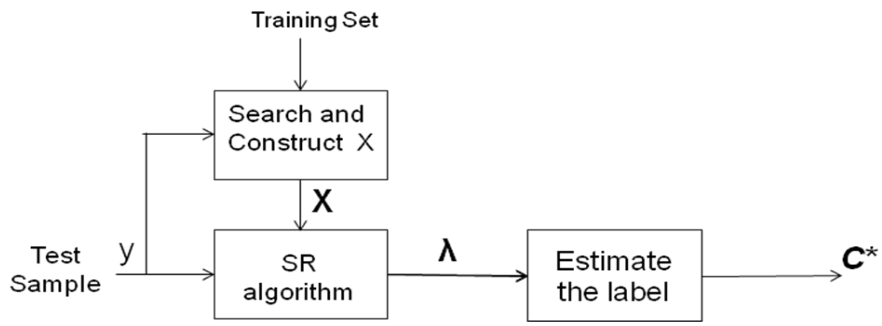



**Fig. 3. General SR approach for phoneme classification**

2.2 SL0 algorithm

Smoothed $l^0$ norm algorithm (SL0) (Mohimani et al., 2009) is an approach to solve the optimization problem (2) without relaxing the $l^0$ norm by $l^1$ or $l^2$ norm. Instead, the $l^0$ norm term $\|\boldsymbol{\lambda}\|_0$ of (2) is substituted by a continuous function of $\boldsymbol{\lambda}$ that approximates the $l^0$ norm. In this approach, (2) is substituted by the following equation:

$$\boldsymbol{\lambda}^* = ArgMin \quad m - \mathrm{F}_\sigma(\boldsymbol{\lambda}) \quad s.t. \ \ to \quad \mathbf{y} = \mathbf{X}.\boldsymbol{\lambda} \tag{6}$$

or equivalently:

$$\boldsymbol{\lambda}^* = ArgMax \quad \mathrm{F}_\sigma(\boldsymbol{\lambda}) \quad s.t. \ \ to \quad \mathbf{y} = \mathbf{X}.\boldsymbol{\lambda} \tag{7}$$

In the above formulation, $\mathrm{F}_\sigma(\boldsymbol{\lambda})$ is a smooth differentiable function of $\boldsymbol{\lambda}$ as an approximation of $m$-$\|\boldsymbol{\lambda}\|_0$. The following equation is an example of $\mathrm{F}_\sigma(\boldsymbol{\lambda})$ (Mohimani et al., 2009):

$$F_\sigma(\lambda) = \sum_{k=1}^{m} \exp(-\lambda_k^2 / 2\sigma^2) \tag{8}$$

where $\sigma$ is the smoothness controlling parameter which affects the accuracy of the approximation. For large $\sigma$, the function is very smooth and its maximization is easy, however, it is not a good approximation of $l^0$ norm. In contrast, small $\sigma$ makes the function a better approximation to $l^0$ norm, but there are many local maxima in the cost function which causes the optimization unreliable. To overcome this deficiency, hoping to escape from getting trapped into local maxima, SL0 decreases $\sigma$ gradually from large values to small values (Mohimani et al., 2009). This approach is called Graduated Non-Convexity (GNC) to optimize a non-convex function (Blake and Zisserman, 1987). If $\sigma = \infty$, the solution of (7) corresponds to the $l^2$ norm solution of equation $\mathbf{y} = \mathbf{X}\,.\,\boldsymbol{\lambda}$ which is a rough solution of (6) and therefore it is a good initial value for $\boldsymbol{\lambda}$ (Mohimani et al., 2009). After assigning the initial value of $\boldsymbol{\lambda}$ and a suitable decreasing sequence for $\sigma$, the maximum of $F_\sigma(\boldsymbol{\lambda})$ will be searched among the set $\{ \boldsymbol{\lambda} \mid \mathbf{y} = \mathbf{X}\,.\,\boldsymbol{\lambda} \}$ by steepest ascent (SA) algorithm. Then the solution is projected to this set. The inner loop of SL0 algorithm is repeated by a fixed, small number of times (L). In other words, to increase the speed, the algorithm does not wait for the steepest ascent algorithm to converge. This may be justified by the gradual decrease in the value of $\sigma$ and the fact that for each value of $\sigma$, it does not need the exact maximize of $F_\sigma$. But, it needs to enter the region near the global maximum of $F_\sigma$ to escape from its local maximum (Mohimani et al., 2009). *It should be noted that the initial point of SA in each step is the maximum of the previous step. Thus, in each step, a better estimation of* $\boldsymbol{\lambda}$ *is obtained*. In Figure 4, the SL0 pseudo code is presented.

- *Initialization*

  1) $\boldsymbol{\lambda}_0 = l^2$ *norm solution of* $\mathbf{y} = \mathbf{X}.\boldsymbol{\lambda}$

  2) *choose a suitable decreasing sequence* $\sigma = [\sigma_1, \dots, \sigma_P]$
- *Main loop*

  *for* $j = 1, \dots, P$

  $\sigma = \sigma_j$

  $\boldsymbol{\lambda} = \boldsymbol{\lambda}_{j-1}$

  $\mu_\sigma = \mu\sigma^2$

  *for* $k = 1, \dots, L$

  $\boldsymbol{\lambda} = \boldsymbol{\lambda} + \mu_\sigma \nabla F_\sigma(\boldsymbol{\lambda})$ *(SA)*

  $\boldsymbol{\lambda} = \boldsymbol{\lambda} - \mathbf{X}^\mathbf{T}(\mathbf{X}\mathbf{X}^\mathbf{T})^{-1}(\mathbf{X}.\boldsymbol{\lambda} - \mathbf{y})$ *(Projection)*

  *end*

  $\boldsymbol{\lambda}_j = \boldsymbol{\lambda}$

  *end*
- *The final answer is* $\boldsymbol{\lambda} = \boldsymbol{\lambda}_P$

**Fig. 4. SL0 pseudo code**

SL0 has been shown to be a fast algorithm with an acceptable accuracy. Its complexity is $O(m^2)$ (for one iteration of the main loop) and may be reduced to $O(m^{1.376})$ by using MSL0 (Mohimani et al., 2010). The evaluation of this representation has been shown to be comparable (and even better in some problems) to LASSO with $O(mn^2)$ and its descendants like Relaxed LASSO with $O(mn^3)$ (Meinshausen 2007) and ABCS with $O(mn^2)$ complexity or reduced complexity ABCS with $O(mn)$ complexity (Sainath et al., 2011b). Therefore, due to reasonable complexity of SL0, It is expected that SL0 would be a good candidate for LVCSR speech recognition applications.

## 3. SPARSE REPRESENTATION BASED N BEST CLASS SELECTOR (SR-NBS)

### 3.1 Motivation

In the sparse classifier of Figure 3, the use a complex SR algorithm (like ABCS or LASSO) results in good accuracy, but greatly sacrifices the classifier speed. On the other hand, by using a fast SR algorithm (e.g. SL0 or similar algorithms) the accuracy is affected, but the classifier becomes fast. Therefore, there is a trade-off between speed and accuracy. This problem arises especially when we need a fast and accurate classifier (e.g. LVCSR). Here, we considered a number of motivations.

Firstly, preliminary experiments and tests on the sparse classifier of Figure 3 showed us that although the accuracy of SL0 based phoneme classifier was not acceptable, it was very fast. Analyzing the results of simulations, we observed that for error classified samples, the correct class was usually located in the second to fifth ranks of the maximization list of (1). Therefore, if we use this algorithm as a class selector, not as a classifier, the number of classes can be reduced and the classification may be performed in a few most probable classes.

Secondly, large margin classifiers like SVM are regarded as the minimum risk classifier in the binary case (Vapnik, 1998). In contrast, this is not true in multiclass case (Vapnik, 1998). In addition, some large margin online learning algorithms (e.g. Passive Aggressive (PA) algorithms (Crammer et al., 2006)) have been originally presented as online learning binary classifiers.

Thirdly, when the training samples are near the test sample, the classifier is expected to be more accurate, because the training samples describe the space of features near the test sample more accurately in this case.

Therefore it seems that SR, instead of using as a classifier, may be employed as an N-best class selector to limit the classifier into certain classes. In addition, a tree search strategy may be used to select the most similar training subset to the test utterance to select the appropriate training subset for each test sample. By using this approach, the secondary classifier may be trained by a limited number of training data that are adapted to the current test example. As a result, test samples can be classified with a better accuracy and with an acceptable complexity. The training procedure of the secondary classifier is repeated for each test sample. However, due to small number of selected neighboring samples, the training would be very fast.

### 3.2 Mathematical formulation

Consider the training set **S** that contains $p$ training samples. Each training sample has a label from the set of class labels **C** with $q$ elements. In speech recognition problems, we usually have $p>>q$ (even $p\rightarrow\infty$). At the first stage, a small subset $\mathbf{S}_y$ with $m$ samples ($m<<p$) is selected using a search strategy at the neighborhood of each test sample $y$. Therefore, as depicted in Figure 5, the training set **S** is reduced to a smaller subset, containing the training samples at the neighborhood of the test sample. This can be shown as:

$$(\mathrm{S}, \mathrm{C}, y) \longrightarrow (\mathrm{S}_y, \mathrm{C}_y, y) \tag{9}$$

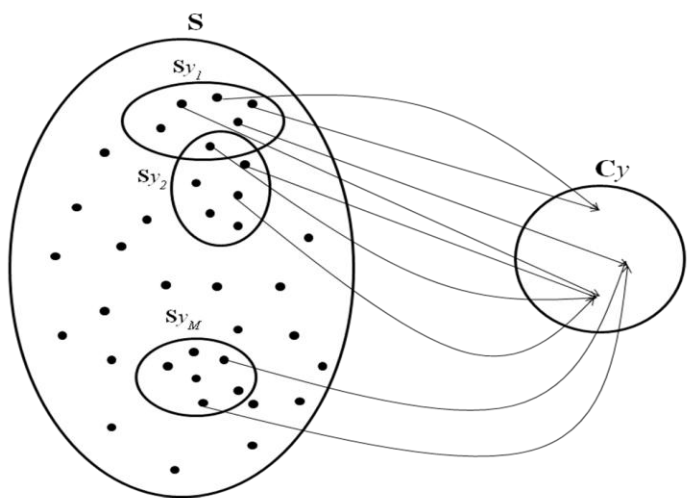



**Fig. 5. The first stage of SR classifiers: selecting small subset of S$_y$ corresponding to the test sample *y***

In common sparse classifiers, the decision is made by applying the SR algorithm and classification rule on (9):

$$(\mathrm{S}_y, \mathrm{C}_y, y) \longrightarrow c^* \tag{10}$$

In contrast, in the proposed classifier (SR-NBS) each subset $\mathbf{S}_y$ is divided into smaller subsets with one corresponding label $c_k$ $(1 \leq k \leq q)$ by a fast SR algorithm (SL0), as depicted in Figure 6. Then N best subsets $\mathbf{S}^1{}_y$ , $\mathbf{S}^2{}_y$ , ... , $\mathbf{S}^N{}_y$ (i.e. the subsets that their corresponding classes are located at the top ranked list of the classifier) are selected. The parameter N was empirically chosen to include the correct label and should be as small as possible.

At this reduced space, we expect that a large margin discriminative classifier can decide fast and with acceptable accuracy. Therefore at the final stage of SR-NBS classifier we have:

$$(\mathrm{S}^1_y, c_1), (\mathrm{S}^2_y, c_2), ..., (\mathrm{S}^N_y, c_N), y \longrightarrow (\text{a large margin model}, y) \longrightarrow c^* \tag{11}$$

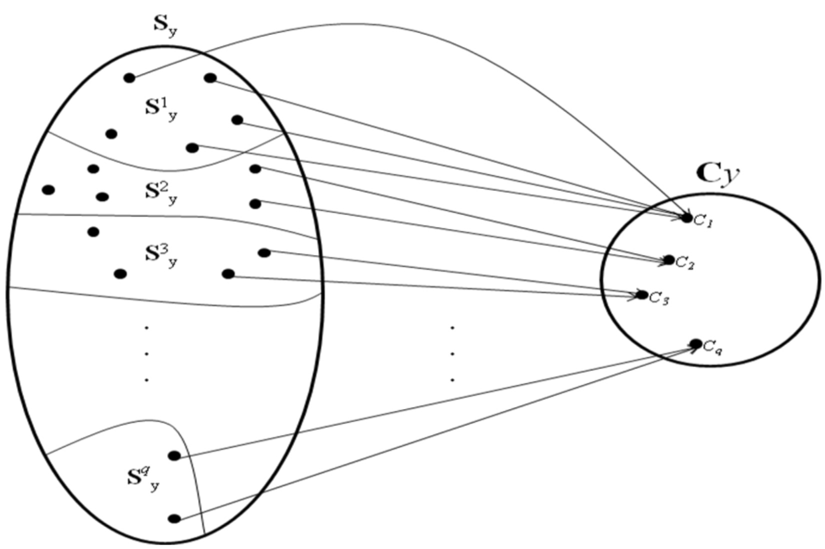



**Fig. 6. The second stage of the proposed classifier (SR-NBS): each subset $\mathrm{S}_y$ is divided into smaller subsets $\mathrm{S}^1{}_y$ , $\mathrm{S}^2{}_y$ , ... , $\mathrm{S}^N{}_y$**

### 3.3 Architecture

The architecture of the proposed phoneme classifier algorithm is depicted in Fig. 6. Firstly, the training selected subset **X** in (6) should be constructed. This is performed by choosing *m* neighbors of the test vector from the training set, by a simple KD-tree fast search algorithm (Berg et al., 2008). After constructing the set **X**, (6) is solved using SL0 algorithm and *N*-best classes are chosen. The *N*-best classes are used to select the proper training set. Finally a discriminative exemplar-based classifier is trained on exemplars of *N*-best classes to determine the final decision on the label of the test sample. In this paper both SVM and PA algorithms are used as large margin discriminative exemplar-based classifiers. Applicability of an online learning algorithm (PA) as the secondary classifier makes the approach flexible for gradual adaptation of the model to test samples in future studies.

### 3.4 Computational complexity

One of the main advantages of this approach is the low complexity of the classifier in the classification stage. Obviously, the system does not employ any offline training (except KD-Tree tree construction). Therefore, the complexity of the approach should be analyzed in the classification phase.

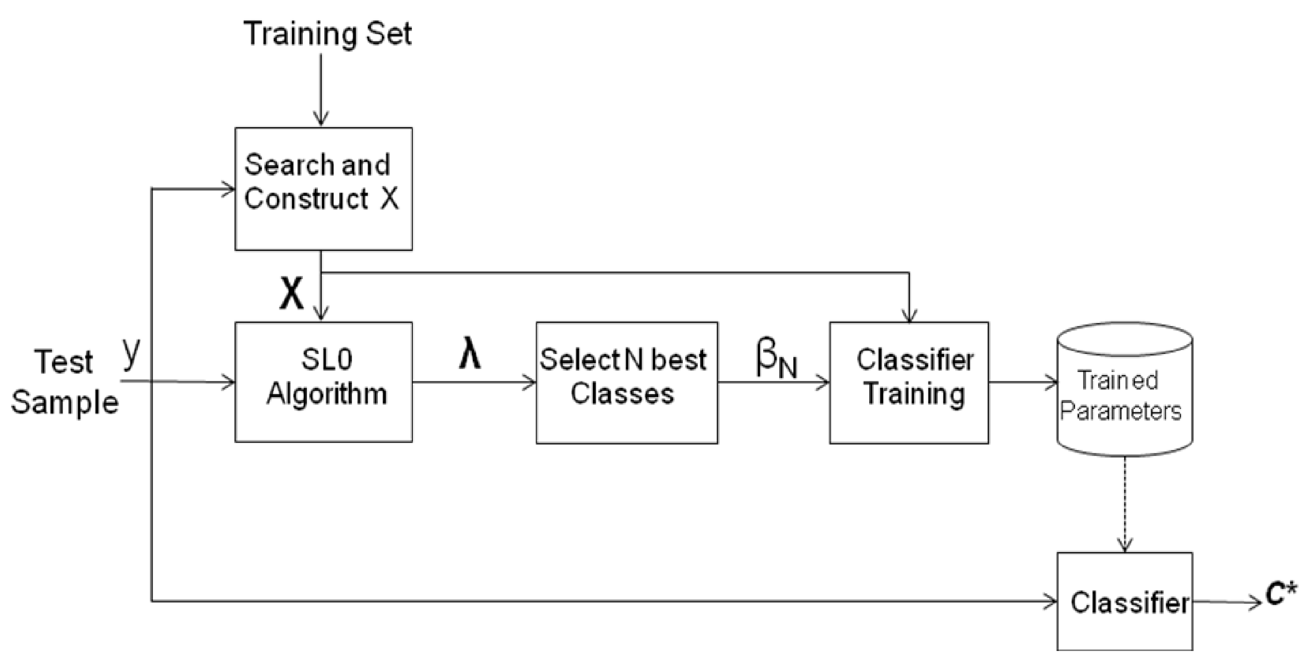



**Fig. 7. Block diagram of SR-NBS classifier**

SR-NBS complexity can be summarized as follows:

- Searching the training set **S** with $p$ samples and selecting $m$ samples from this set. We used KD-tree query and then KNN search to select $m$ best samples. As the tree can be built off-line, the search time on Kd-tree is only the query time that its computational order is $O(p^{1-1/n}+m_t)$ where $m_t$ is the number of samples that are reported by the query on KD-tree search and $n$ is the dimension of samples (Berg et al., 2008).Then $m$ samples should be selected from $m_t$ samples by KNN search. The complexity of this exhaustive search is $O(nmm_t)$ (Chen et al., 2000).
- SL0 algorithm. For accurate complexity analysis of SL0, we used the pseudo code, which is presented in Fig. 3. The computationally expensive parts of the algorithm are two parts. First, calculating the term $\mathbf{X}^{\mathrm{T}}(\mathbf{XX}^{\mathrm{T}})^{-1}$ out of the main loop one time per each sample. It needs two matrix multiplication each with $O(n^2m)$ complexity and one matrix inversion with roughly $O(n^3)$ complexity (Strang, 2003). Therefore its total computational complexity is roughly $O(2n^2m+n^3)$. The second computationally complex part of SL0 algorithm is calculating the term $\boldsymbol{X}^T(\boldsymbol{XX}^T)^{-1}(\boldsymbol{X}\ .\ \boldsymbol{\lambda}-\boldsymbol{y})$ inside the main loop which needs two matrix by vector multiplication each with $O(nm)$ complexity. Therefore its total computational complexity is roughly $O(2dnm)$ where $d$ is the number of the iterations.
- Applying second classifier in the reduced space. This classifier should be initially trained and then the test sample should be classified. As there are a few samples in the reduced space, this classifier can be trained and employed to classify the test sample very fast. For example, if we use SVM classifier with $r$ training samples and $\alpha$ support vectors and $N_c$ binary classifiers (for multi-class classification problems), a reasonable training complexity is between $O(N_cr^2)$ and $O(N_cr^3)$ and the test complexity is between $O(N_c\alpha)$ and $O(N_c\alpha^2)$ (Tang et al., 2009; Basu et al., 2003).

The results of computational complexity analysis of SR-NBS are shown in Table 1. By assuming $p$=142879, $n$=40 and $m_t$=1000[1], $m$=200, $d$=15, $N_c$=1 (binary SVM), $r$=80, and $\alpha$=60[2], it is observed that the computational complexity of SR-NBS

---

[1] It should be noted that the exact value of $m_t$ is not specified. In our experiments it depends on the chosen threshold for the KD-tree search.

[2] In reduced space, we have $\alpha \leq r \leq m$=200.

classifier is dominated by KNN search to select *m* neighbors of the test sample from $m_t$ samples (resulted from KD query).

**Table 1. Approximate computational complexity of SR-NBS classifier**

| SR-NBS step | computational complexity |
|---|---|
| KD query | $O(p^{1-1/n} + m_t)$ |
| KNN search | $O(nm\ m_t)$ |
| SL0 (out of the main loop) | $O(2n^2m+n^3)$ |
| SL0 (inside the main loop) | $O(2dnm)$ |
| SVM training | between $O(N_c r^2)$ and $O(N_c r^3)$ |
| SVM classify | between $O(N_c \alpha)$ and $O(N_c \alpha^2)$ |

*p*: training samples   *n:* dimension of samples   $m_t$ : samples results from query   *m:* KNN output samples
*d* : SL0 iterations   $N_c$ : the number of SVM binary classifiers   *r*: SVM training samples   $\alpha$: support vectors

It is worth to compare the SR-NBS complexity with other well-known classifiers. If we use KNN classifier, finding out *k* neighbors of each test sample by exhaustive search, its computational complexity is $O(kpn)$. By assuming $p$=142879, $k$=22 and $n$=40, we conclude that SR-NBS is very fast in comparison to KNN classifier. On the other hand, if we use a general SVM classifier with the test complexity between $O(N_c\alpha)$ and $O(N_c\alpha^2)$, assuming that about %50 of 142879 training samples have been selected as the support vectors, then we have $\alpha$=70000. If the one versus one strategy is used for multiclass classification, and the number of the classes is 48 (i.e. for phoneme classification), we have $N_c$=(48*47)/2=1128. Therefore the computational complexity of SVM classifier is much higher that SR-NBS classifier.

# 4. Experiments and Results

## 4.1 Evaluation Benchmark

To assess the proposed classification approach, a set of experiments were conducted on extracted features from TIMIT database (Lemel et al., 1986). TIMIT contains phonetically balanced 6300 sentences where 10 sentences are uttered by each of 630 speakers from 8 major dialect regions of the United States. In this paper, 3696 utterances from training set and core test set containing 192 utterances from test set were used as training and evaluation sets, respectively. The test was employed in accordance with standard examinations on TIMIT (Lee and Hon, 1989). Firstly, 61 phonetic labels were converted to 48 labels. Then, the acoustic model was trained with 48 labels (Lee and Hon, 1989). Finally, these 48 labels were collapsed into a smaller set of 39 labels to improve the recognition performance (Lee and Hon, 1989). The segmental features were extracted as in (Gao et al., 2001). At the first stage, each speech utterance was chunked into 20ms frames with a frame shift of 5ms.Then, 13 Mel frequency cepstral coefficients (MFCC) of each frame were extracted. The MFCC vectors of beginning, middle and ending frames of each phoneme were averaged and the resulting three vectors were merged. Therefore, a 39 dimensional vector was obtained. Then, a 117 dimensional vector per each three consecutive phonemes was generated. Finally, the dimensionality of this vector was reduced to 40 by linear discriminative analysis (LDA) transform[3]; perhaps to deal with curse of dimensionality effect.[4] By this approach, 3696 and 192 training and test utterances were converted to 142879 and 7330 vectors respectively where each vector represented one phoneme. The experiments were evaluated based on the architecture of Fig. 6. The number of neighbors in KD-Tree search was set to 200 (Sainath et al., 2010).[5]

### 4.2 Parameter selection of SR-NBS classifier

**Experiment 1.** In the first experiment, the probability that the label of test sample is in the N-best class list (the best selected classes by SL0 algorithm) was investigated.[6] As shown in Fig. 7, the test samples were located at 3 to 5 best classes with the probability 0.9 to 0.95 respectively. Therefore, it seems that only 5 best classes are sufficient to select and use at the secondary classifier.

**Experiment 2.** In the next experiment, the test set accuracy of the proposed architecture using an SVM classifier with Radial Basis Function (RBF) kernel (Vapnik, 1998) and by applying the KD-tree selected examples of the SL0 selected classes as the training set was evaluated.[7] Firstly, the suitable parameters for the SVM kernel (C and Gamma) and SL0 algorithm were chosen by a grid search. As shown in Fig. 8, the best accuracy was achieved when the number of selected classes was two. It means that although the correct label of 83% of test samples were located at 2-best classes in comparison to nearly 96% in the case of 5 best classes, the SVM classifier classified 86% of them correctly and the accuracy of

---

[3] For LDA transform we used the MATLAB code available at http://homepage.tudelft.nl/19j49.

[4] Although, LDA is theoretically justified for jointly Gaussian observations, but the advantage of using LDA for dimensionality reduction of features has been revealed experimentally in other natural conditions, including speech recognition. For example see (Eisele et al., 1996), (Gao et al., 2001) and (Zolnay et al., 2005).

[5] For KD search algorithm we used the MATLAB code available at http://guy.shechter.org

[6] For SL0 algorithm we used the MATLAB code available at http://ee.sharif.ir/~SLzero

[6] For SVM algorithm we used the LIBSVM toolbox available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

71.33% was obtained. In contrast, in the case of 5-best classes, only 70% of test samples were classified correctly. Therefore, it is better to use a 2-best class candidates set for the final classifier training when the second classifier is SVM. It should be noted that the best selection for N may depend on the second classifier architecture. Therefore, if another classifier is used instead of SVM, the optimum value of N may be different.

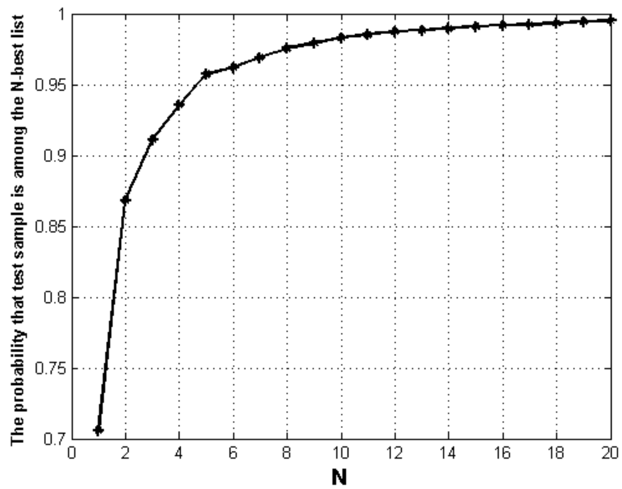



**Fig. 8. The probability that test sample is among the N-best list versus N**

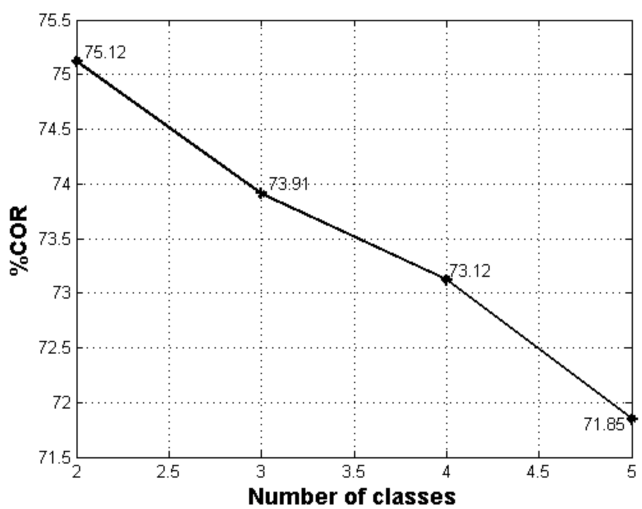



**Fig. 9. Percentage of correct classification of SR-NBS phoneme classifier versus the number of best classes**

## 4.3 Empirical comparison of accuracy and computational complexity of SR-NBS with other classifiers

In this experiment, the accuracy and computational complexity of SR-NBS classifier was compared empirically with some well known classifiers while they were used as the phoneme classifier. The parameters of all algorithms (except to SVM2) were selected by grid search to empirically optimize the error rate. For SVM2, the parameters were selected to result in a small number of support vectors and to achieve maximum speed. In this experiment, online learning PA and SVM classifiers with RBF kernel (C=1.7, Gamma=0.044) were used as the secondary classifier and the result of 2-best class candidates was used to train this final classifier.[8] As indicated in Table 2, SR-NBS-SVM and SR-NBS-PA classifiers both outperform the well known classifiers like KNN with K=22, PA with RBF kernel (C=0.7, Gamma=0.02), SR-SL0 and SVM2 with RBF kernel ($C=2*10^7$, $Gamma=2*10^{-8}$), resulting a moderate number of support vectors (52.3% of training samples). However, SVM1 classifier with RBF kernel (C=1.7, Gamma=0.04) and large number of support vectors (74.5% of training samples) has better accuracy in comparison to SR-NBS-SVM and SR-NBS-PA classifiers. The latter classifier suffers from very high complexity and is not an appropriate candidate for most ASR applications. It should be mentioned that we have reported the mean of the accuracy for proposed algorithms in Table 2. As some software packages, which we used in our algorithms, are sensitive to the order of the training samples, the accuracy could change slightly. For example, the unbiased standard deviation of accuracy for SR-NBS-PA and SR-NBS-SVM are 0.207 and 0.26 respectively for 10 independent evaluations.

**Table 2. Accuracy of different phoneme classifiers on TIMIT**

| classifier | accuracy (percent) |
|---|---|
| SVM1[1] | 77.09 |
| SVM2[2] | 71.30 |
| KNN | 73.76 |
| PA[3] | 72.00 |
| SR-SL0 | 70.60 |
| SR-NBS-PA | 74.25 |
| SR-NBS-SVM | 75.12 |

[1]RBF kernel (C=1.7, Gamma=0.04) and 74.5% of training samples were selected as the support vector
[2]RBF kernel ($C=2*10^7$, $Gamma=2*10^{-8}$) and 52.3% of training samples were selected as the support vector
[3]RBF kernel (C=0.7, Gamma=0.02) and 74.8% of training samples were selected as the support vector

[8] For PA algorithm we used the MATLAB code available at http://dogma.sourceforge.net.

**Statistical significance of the results:** As we used a large number of samples in our experiments (7330 test vectors), the reported results in Table 2 are correct with probability more than 0.99. For example, to calculate the statistical significance of the results for SR-NBS-SVM and KNN classifiers, we have obtained from simulations that 5507 and 5370 test vectors were classified correctly by SR-NBS-SVM and KNN respectively. On the other hand, 1824 and 1960 test vectors have been classified incorrectly by SR-NBS-SVM and KNN respectively. Therefore the group means ($\mu_1$ and $\mu_2$) and the overall mean ($\mu$ ) are:

$$\mu_1 = 5506 \times 100 + 1824 \times 0 = 75.12$$
$$\mu_2 = 5370 \times 100 + 1960 \times 0 = 73.26$$
$$\mu = (75.12 + 73.26)/2 = 74.19$$

The sum of squares of errors ($SSE$) and sum of squares between the groups ($SSG$) are:

$$SSE = 5506 \times (100 - 75.12)^2 + 1824 \times (0 - 75.12)^2 + 5370 \times (100 - 73.26)^2 + 1960 \times (0 - 73.26)^2 \approx 27{,}976{,}037$$
$$SSG = 5506 \times (100 - 74.19)^2 + 1824 \times (0 - 74.19)^2 + 5370 \times (100 - 74.19)^2 + 1960 \times (0 - 74.19)^2 \approx 28{,}072{,}840$$

As the degree of freedom for individual within groups ($DFE$) and between groups ($DFG$) are:

$$DFE = 7330 - 2 = 7328$$
$$DFG = 2 - 1 = 1$$

Therefore mean square error ($MSE$), mean square for groups ($MSG$) and F-Test parameter ($F$ ) are:

$$MSE = SSE / DFE \approx 3817.7$$
$$MSG = SSG / DFG \approx 28{,}072{,}840$$
$$F = MSG / MSE \approx 7353.338$$

By consulting an F-distribution table with $DFE = 7328$ and $DFG = 1$ we find that the probability of $F = 7353.338$ is less than 0.01. Therefore, the achieved results are statistically significant with the probability more than 0.99.

Although, the mathematical analysis and analytical calculation of the computational complexity of the proposed phoneme classifier and comparing it with some well-known classifiers have been presented in section 3.4 and Table 1 which shows that this algorithm is faster than them, but it is informative to compare the complexities empirically. Therefore, we used a laptop computer with Intel core I5 2.53GHZ CPU and 4GB RAM and well-known MATLAB toolboxes. As the toolboxes which may not be optimized in programming codes, their empirical speed may not be comparable. Therefore, as we have used PA and SVM classifiers in the final stage of the proposed approach, we compared the empirical complexity of the proposed phoneme classifier with PA and SVM classifiers. Tables 3 and 4 compare the time complexities of SR-NBS-PA and SR-NBS-SVM with PA and SVM classifiers respectively. As it is indicated in Tables 3 and 4, excluding the KD search, our approach makes the classification faster and with less time complexity than PA and SVM classifiers. It is because our classifier firstly searches and selects a smaller training subset and then trains the classifier with this subset. Therefore, the number of the support vectors is much smaller than SVM or PA which selects the support vectors from the whole training set. A closer study of SR-NBS-SVM and SR-NBS-PA classifiers in Tables 2, 3 and 4 indicates that SR-NBS-SVM is faster and more accurate, comparing to SR-NBS-PA. It is because PA is an online learning algorithm; therefore its training is slower than SVM training for small subset of training samples.

For a better understanding of the speed of the algorithm, we compared our proposed phoneme classifier with the sparse representation phoneme recognition system which was proposed in (Sainath et al., 2011a). The authors computed the average time per frame to search the training subset and to estimate the label of each frame. They reported that the frame classification time is approximately 3 times of the frame search time. But as indicated in Table 3 and 4, in our proposed approach (especially for SR-NBS-SVM) the classification time is much less than the search time.

**Table 3. Time complexity (msec per test sample) for two PA based phoneme classifiers on TIMIT**

| Classifier | search (msec) | classify (msec) | total test (msec) |
|---|---|---|---|
| PA | ----- | 68.6 | 68.6 |
| SR-NBS-PA | 92.4 | 44.1 | 136.5 |

**Table 4. Time complexity (msec per test sample) for three SVM based phoneme classifiers on TIMIT**

| Classifier | search (msec) | classify (msec) | total test (msec) |
|---|---|---|---|
| SVM1 | ----- | 395.1 | 395.1 |
| SVM2 | ----- | 312.8 | 312.8 |
| SR-NBS-SVM | 92.4 | 4.4 | 96.8 |

Finally we compared the proposed phoneme classifier with some reported phoneme classifiers on TIMIT and summarized the results in Table 5. As it can be observed in Table 5, the accuracy of the proposed phoneme classifier is better than most of the proposed phoneme classifier. It is better than hierarchical phoneme classifiers (Dekel et al., 2005), some combinational phoneme classifiers like HMM-GMM, HMM-ANN and HMM-MLP (Pinto and Hermansky, 2008) and GMM and KNN (Sainath et al., 2010). On the other hand, SVM and CS-$H_{lin}H^2H^3$ classifiers (Sainath et al., 2010) have better accuracy in comparison with SR-NBS-SVM and SR-NBS-PA. As stated above, these two phoneme classifier are much more complex than our proposed classifiers.

**Table 5. The accuracy comparison of the proposed approach with some reported phoneme classifiers on TIMIT**

| classifier | accuracy (percent) |
|---|---|
| Online Hierarchical (Tree)[1] | 60.00 |
| Online Hierarchical (Flat)[1] | 61.30 |
| Batch Hierarchical (Tree)[1] | 59.40 |
| Batch Hierarchical (Flat)[1] | 58.20 |
| Batch Hierarchical (Greedy)[1] | 41.80 |
| HMM-GMM (3-states)[2] | 64.10 |
| Hierarchy, GMM posteriors(3-states)[2] | 68.40 |
| Hierarchy, GMM log-likelihoods(3-states)[2] | 71.00 |
| HMM-ANN (3-states)[2] | 71.60 |
| Hierarchy, MLP posteriors(3-states)[2] | 73.40 |
| GMM[3] | 74.19 |
| KNN[3] | 73.69 |
| SVM[3] | 76.20 |
| CS-$H_{lin}H^2H^3$ [3] | 76.44 |
| **SR-NBS-PA** | **74.25** |
| **SR-NBS-SVM** | **75.12** |

[1] (Dekel et al., 2005)
[2] (Pinto and Hermansky, 2008)
[3] (Sainath et al., 2010)

### 4.4 SR-NBS in noisy environment

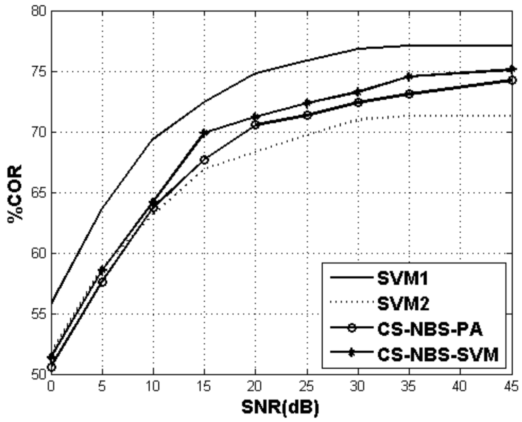
In the next experiments, the accuracy of SR-NBS phoneme classifier was compared to SVM classifier in additive noise environment. We used different additive noise typed driven from NoiseX database (Varga et al., 1992) and we compared SR-NBS-PA and SR-NBS-SVM classifiers with two SVM classifiers with the parameters which are mentioned in Table 2. Other test conditions were as in section 4.1. The noise was added to the first 3696 and 192 training and test utterances and 142879 and 7330 vectors were extracted respectively.

**Experiment 1.** In this experiment, the accuracies of the classifiers were compared in matched training condition. The classifiers were trained and tested with noisy vectors with the same signal to noise ratios using white noise. The results are shown in Figure (10-a). As shown in this figure, the accuracies of SR-NBs classifiers were between SVM1 and SVM2 classifiers like noiseless condition (see Table 2). In addition, the accuracy of SR-NBS-SVM outperforms SR-NBS-PA.

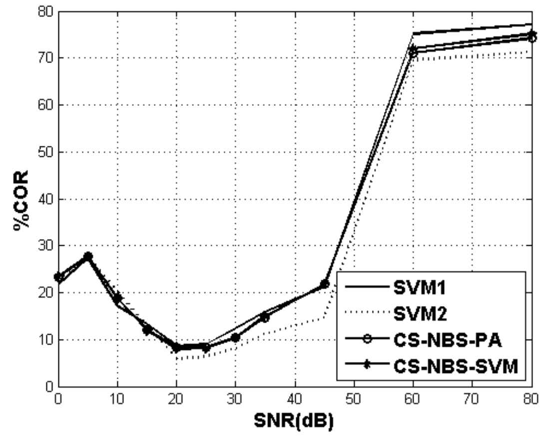
**Experiment 2.** In this experiment, the accuracies of the classifiers were compared in clean training condition. Therefore the classifiers were trained with noiseless training vectors. Then, they were tested by noisy test vectors using white noise. The results are shown in Figure (10-b). As shown in this Figure, the results of all classifiers were almost the same.

**Experiment 3.** In this experiment, the accuracies of the classifiers were compared in a general training condition using white noise. In this case, the classifiers were trained with noisy training vectors with various signal to noise ratios. Then, they were tested by test vectors with a fixed, but unknown signal to noise ratio. The results are shown in figure (10-c). As shown in this Figure, the accuracies of SR-NBS classifiers were between SVM1 and SVM2 classifiers and the accuracies of SR-NBS-SVM and SR-NBS-PA were the same.

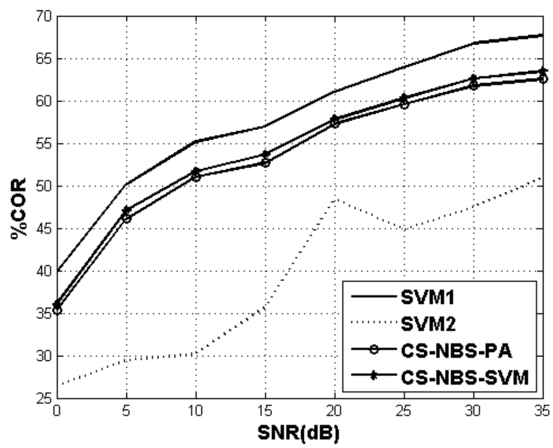
**Experiment 4.** In this experiment, the accuracies of SR-NBS-PA and SR-NBS-SVM were evaluated in other types of noisy environments. We added white, babble, exhibition and office additive noise from NoiseX database to each training and test utterance of TIMIT in 20dB signal to noise ratio. We also corrupted the utterances by AR model reported in (Singh and Chaterjee, 2011) in 20dB signal to noise ratio. The results are shown in Table 6. We observe that both proposed phoneme classifiers are robust to additive noise characteristics and also interferences resulted from the mentioned AR model.




**(a)**




**(b)**




**(c)**

**Fig. 10. Accuracy comparison of the classifiers in additive white noise environment (a) matched training (b) clean training (c) general training**

**Table 6. Accuracy comparison of SR-NBS-SVM and SR-NBS-PA phoneme classifiers in various noisy environments for S/N=20dB and match training**

| classifier | %accuracy | | | | |
|---|---|---|---|---|---|
| | White | Babble | Exhibition | Office | AR model |
| SR-NBS-PA | 70.53 | 70.41 | 70.46 | 69.74 | 70.16 |
| SR-NBS-SVM | 71.19 | 71.10 | 71.32 | 70.12 | 71.06 |

**Experiment 5.** In the final experiment, we compared the accuracies of SR-NBS-SVM and SVM1 (SVM with better accuracy and more support vectors) classifiers in an almost equal computational complexity. Therefore, we used 1428790 and 220000 training set which contained the training vectors with various signal to noise ratios for SR-NBS-SVM and SVM1 classifiers respectively. In this condition the computational complexities of these two classifiers were almost the same. The accuracies of the classifiers were compared in 10dB signal to noise. The results are shown in Table 7 which indicates that SR-NBS-SVM classifier has was more accurate in comparison with SVM1 classifier in approximately the same computational complexity.

**Table 7. Accuracy comparison of SR-NBS-SVM and SVM1 phoneme classifiers for approximately same computational complexity and S/N=10dB**

| classifier | %accuracy |
|---|---|
| SVM1 | 57.18 |
| SR-NBS-SVM | 59.84 |

## 5. CONCLUSIONS AND FUTURE WORKS

In this paper, SR-NBS classifier has been introduced which is a fast phoneme classifier with acceptable accuracy. The key contribution of the paper is the reduction of the number of classes (labels) using a fast SR model; therefore, the classifier encounters with a reduced number of more probable labels and can decide better and more efficiently. This procedure was implemented by a search algorithm and then a fast SR algorithm. Finally a well-known classifier was trained by these reduced and adapted training samples. Simulation results showed that SR-NBS is very fast and accurate enough in clean and noisy environments. In addition, SR-NBS has a good potential to increase the speed and accuracy. Making this trade-off between these two criteria is very promising.

Our future works will be directed in three subjects. At first, we decide to investigate some faster searching algorithms to increase the SR-NBS speed while maintaining its accuracy. Specially as we used $l^0$ norm SR algorithm and unlike the $l^1$ norm SR algorithms it can be modified to a weighted sparse signal decomposition (Babaie-Zadeh et al., 2012), we expect that by implementing this algorithm, the search in SR-NBS becomes faster. Secondly, we are trying to use this phoneme classifier as a recognizer engine for ASR and then extend it to LVCSR applications. Finally, the study can be more extended to adapt the classifier parameters to the test sample.

REFRENCES

Amrouche, A., Debyeche, M., Taleb-Ahmed, A., Rouvaen, J.M., Yagoub, M., 2010.An efficient speech recognition system in adverse conditions using the nonparametric regression. Eng. Appl. Artif. Intell. 23, 85–91.

Asaei, A., Taghizadeh, M.J., Bourlard, H., Cevher, V., 2011. Multi-party speech recovery exploiting structured sparsity models. proc. 12th Annual conf. Inter. Speech Communication Association (Interspeech2011), 185-188.

Babaie-Zadeh, M., Mehrdad, B., Giannakis, G.B., 2012. Weighted sparse signal decomposition. proc. int. conf. Acoust. Speech Signal Process. (ICASSP), 3425-3438.

Basu, A., Watters, C., Shepherd, M., 2003. Support Vector Machines for text categorization. proc. 36th Hawaii int. conf. on syst. sci. (HICSS), 103-109.

Berg, M.D., Cheong, O., Kreveld, M.V., Overmars, M., 2008. Computational Geometry: Algorithms and Applications, third ed., Springer.

Blake A., Zisserman, A., 1987. Visual Reconstruction, MIT Press.

Carmi, A., Gurfil, P., Kanevsky, D., Ramabhadran, B., 2009. ABCS: Approximate Bayesian Compressed Sensing. IBM Technical Report, Human Language Technologies.

Chen, S.S., Donoho, D. L., Saunders, M. A., 1999. Atomic decomposition by basis pursuit. SIAM J. Sci. Comput. 20(1), 33-61.

Chen, Y.S., Hung, Y.P., Fuh, C.S., 2000. Winner-Update algorithm for nearest neighbor search. proc. 36th International conference on pattern recognition 2, Barcelon, Spain, 708-711.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y., 2006. Online passive aggressive algorithms. J. Mach. Learn. Res. 7, 551-585.

Dekel, O., Keshet, J., Singer, Y., 2004. An online algorithm for hierarchical phoneme classification. Workshop on multimodal Interaction and related Mach. Learn. algorithms (lecture notes in computer science), Springer-Verlag, 146-159.

Donoho, D., 2006. Compressed sensing. IEEE Trans. Inform. Theory 52, 1298-1306.

Eisele, T., Haeb-Umbach, R., Langmann, D., 1996. A comparative study of linear feature transformation techniques for automatic speech recognition. proc. int. conf. on Spoken Lang. (ICSLP 96), 252-255.

Elad, M., 2010. Sparse and redundant representations, Springer Press.

Gao, Y., Li, Y., Goel, V., Picheny, M., 2001. Recent advances in speech recognition system for IBM DARPA communicator. proc. Eurospeech2001, 503-506.

Gemmeke, J.F., Virtanen, T., 2010. Noise robust Exemplar-Based connected digit recognition. proc. int. conf. Acoust. Speech Signal Process. (ICASSP), 4546-4549.

Gemmeke, J.F., Hammen, H.V., Cranen, B., Boves, L., 2010. Compressive sensing for missing data imputation in noise robust speech recognition. IEEE J. Sel. Top. Signal Process. 4(2), 272-287.

Gorodnitsky, I. F., Rao, B.D., 1997. Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm. IEEE Trans. Signal Process. 45(3), 600-616.

Joachims, T., 2002. Learning to classify text using Support Vector Machines: methods, theory and algorithms. Kluwer Academic Publishers, Norwell, MA, USA.

Juang, B., Rabiner, L., 2005. Automatic speech recognition - a brief history of the technology development, 2nd ed., Elsevier Encycl. Lang. and Linguist.

Keshet, J., Shalev-Shwartz, S., Singer, Y., Chazen, D., 2007. A large margin algorithm for speech-to-phoneme and music-to-score alignment. IEEE Trans. Audio, Speech Lang. Process. 15(8), 2373-2382.

Lee, K.F., Hon, H.W., 1989. Speaker independent phone recognition using Hidden Markov Models. IEEE Trans. on Acoust. Speech and Signal Process. 37, 1641-1648.

Lemel, L., Kassel, R., Seneff, S., 1986. Speech database development: Design and analysis of the acoustic phonetic corpus. proc. DARPA Workshop on Speech Recognition, 100-109.

Li, X., Wang, L., Sung, E., 2008.AdaBoost with SVM-based component classifiers. Eng. Appl. Artif. Intell. 21, 785–795.

Meinshausen, N., 2007. Relaxed LASSO. Comput. Stat. Data Anal. 52, 374-393.

Mohimani, H., Babaie-Zadeh, M., Jutten, C., 2009. A fast approach for overcomplete sparse decomposition based on smoothed L0 norm. IEEE Trans. Signal Process. 57(1), 289-301.

Mohimani, H., Babaie-Zadeh, M., Gorodnitsky, M., Jutten, C., 2010. Sparse recovery using smoothed L0 norm (SL0): convergence analysis. Available: {arXiv:cs.IT/1001.5073}.

Ping, L., Chun-Guang, Z., Xu, Z., 2010. Improved support vector clustering. Eng. Appl. Artif. Intell. 23, 552–559.

Pinto, J., Hermansky, H., 2008. Combining evidence from a generative and a discriminative model in phoneme recognition. proc. 10th Inter. conf. on Spoken Lang. Process. (ICSLP2008-Interspeech2008).

Rabiner, L., 1989. A tutorial on Hidden Markov Models and selected applications in speech recognition. Proc. of the IEEE 77(2), 257-286.

Saeb, A., Razzazi, F., 2012. A fast compressive sensing approach for phoneme classification. proc. int. conf. Acoust. Speech Signal Process. (ICASSP), 4281-4284.

Sainath, T.N., Carmi, A., Kanevsky, D., Ramabhadran, B., 2010. Bayesian compressive sensing for phonetic classification. proc. int. conf. Acoust. Speech Signal Process. (ICASSP), 4370-4373.

Sainath, T.N., Ramabhadran, B., Picheny M., Nahamoo, D., Kanevsky, D., 2011a. Exampler-based sparse representation features: from TIMIT to LVCSR. IEEE Trans. Audio, Speech Lang. Process. 19, 2598-2613.

Sainath, T.N., Ramabhadran, B., Nahamoo, D., Kanevsky, D., 2011b. Reduced computational complexities of exemplar-based sparse representation with applications to large vocabulary speech recognition. proc. 12th Annual conf. Inter. Speech Communication Association (Interspeech2011), 785-788.

Seneviratne,A.J., Solo, V., 2012. On vector $L_0$ penalized multivariate regression. proc. int. conf. Acoust. Speech Signal Process. (ICASSP), 3613-3616.

Singh, S.D., Chatterjee, A., 2011. A comparative study of adaptation algorithms for nonlinear system identification based on second order Volterra and bilinear polynomial filters. Measurement 44 (10), 1915-1923.

Soan, G., Zweig, G., Kingsbury, B., Mangu, L., Chaudhari, U., 2003. An architecture for rapid decoding of large vocabulary conversational speech. proc. Eurospeech2003.

Strang, G., 2003. Introduction to Linear Algebra, Wellesley-Cambridge Press.

Tang, Y., Zhang, Y.Q., Chawla, N.W., Krasser, S., 2009. SVMs modeling for highly imbalanced classification. IEEE Trans. Syst. Man and Cybern. Part B:Cybernetics 39(1), 281-288.

Tibshirani, R., 1996. Regression shrinkage and selection via the LASSO. J. Royal Stat. Soc. Ser. B (Methodol.) 58(1), 267-288.

Tipping, M., 2001. Sparse Bayesian and the relevance vector machine. J. Mach. Learn. Res. 1, 211-244.

Tzicas, D., Wei, L., Likas, A., Yang, Y., Galatsanos, N., 2006. A tutorial on Relevance Vector Machines for regression and classification with applications. Eurosip 17(2).

Ulfarsson, M.A., Solo, V., 2011. Vector $L_0$ sparse variable PCA. IEEE Trans. Signal Process. 59(5), 1949-1958.

Ulfarsson, M.A., Solo, V., 2012. Sparse loading noisy PCA using an $L_0$ penalty. proc. int. conf. Acoust. Speech Signal Process. (ICASSP), 3597-3600.

Vapnik, V. N., 1998. Statistical Learning Theory, Wiley Press.

Varga, A., Steeneken, H.J.M., Tomlinson, M., Jones, D., 1992. The Noisex-92 study on the effect of additive noise on automatic speech recognition. Tech. Rep. DRA Speech Res. Unit.

Wang, X.Y., Fu, Z.K., 2010.A wavelet-based image de-noising using least squares support vector machine. Eng. Appl. Artif. Intell. 23, 862–871.

Wright, J., Yang, A., Ganesh, A., Sastry, S.S., Ma, Y., 2009. Robust face recognition via sparse representation. IEEE Trans. Pattern Anal. Mach. Intell. 31, 210-227.

Zolnay, A., Schlüter, R., Ney, H., 2005. Acoustic feature combination for robust speech recognition. proc. int. conf. Acoust. Speech Signal Process. (ICASSP), 457-460.

Zou H., Cole, R., 2005. Regularization and variable selection via the Elastic Net. J. Royal Stat. Soc., 301-320.