ISSN 2047-3338

Efficient Normalization of Features Extracted from Vocal Signal for Support Vector Machine

Boutkhil SIDAOUI[1] and Kaddour SADOUNI[2]

[1]Mathematics and Computer Science Department, University Center Salhi Ahmed- Naâma, Algeria
[2]Computer Science Department, University of Science and Technology Mohamed Boudiaf- Oran, Algeria
[1]b.sidaoui@cuniv-naama.dz, [2]kaddour.sadouni@univ-usto.dz

***Abstract*– In this paper we introduce and investigate the performance of an efficient normalization of features extracted from signals of vowels, into fixed-length features vectors, in the context of vowels classification by kernel machine learning methods. The proposed models capture the speech signal and extract features vectors of the same size, over the entire vowel segment. In this work, we propose three approaches based on signal segmentation to windows, in the extraction of MFCC coefficients. In the aim to produces a compact size, yet wealthy and informative training dataset, also to give state of the art vowels classification results. To improve the performance of the proposals approaches, we are opted to use support vector machine method for vowels classification, due its performance in machine learning. Recognition rates of 58.76%, 58.85% and 61.19 %, on the 20 vowels of TIMIT corpus was achieved by the three approaches, for handling the multi category nature of vowels classification, using SVM method with one-versus-one and one-versus-all strategies. Encouraging results have been achieved with 20 vowels of TIMIT datasets.**

***Index Terms*– Features Extraction, MFCC, Classification and Support Vector Machine Multiclass**

I. INTRODUCTION

AUTOMATIC Speech Recognition (ASR) has been the focus of both the machine learning and speech communities for the past few decades due to its importance in any conceivable man machine interface (MMI). However, because of the large variability in the way humans communicate and the background noise corrupting the speech signal, ASR is considered perhaps as the most difficult and challenging problem to be solved for many years to come. Speech recognition is usually accomplished in two main steps. The first one consists of extracting a set of useful, yet compact, features that capture important characteristics of speech data. It is followed by the recognition step where the features, obtained in the previous one, are fed into a classifier for word prediction in the case of speech recognition, or simply the phonetic class in phonetic recognition. It is difficult to imagine constructing good recognizers with low quality features on one hand and on the other hand, if feature extraction is carried out to perfection, then the patterns are easily distinguishable and consequently, one can obtain good recognition results even with simple and naïve machine learning algorithms.

Moreover, both steps must be performed faster than the speech sampling rate in order for the recognition to be carried out in real time. Most of the successful approaches for speech feature extraction are based on the standard Mel Frequency Cepstral Coefficients (MFCC) extracted for successive and overlapping windows.

Many approaches have been proposed towards the goal of improving the performance of ASR. The first class of such methods [1]–[4] focuses on new techniques for feature extraction. In the second class, keep the MFCC features along with derivative information, but use novel powerful discriminative methods [5]–[7].

Our contribution falls within the third class of methods which use some combination of MFCC along with powerful discriminative methods.

The remainder of the paper is organized as follows. In section I, we introduce this work, in section II, we discuss some related successful methods for feature extraction. In the following section, large margin kernel methods will be briefly discussed for phonetic classification. In section IV we introduce related work of SVM for phonetic classification. In section V, we introduce our contribution feature extraction, i.e., variable window step and variable window size framework, in the next section we give the best results of extensive experiments on the TIMIT data base using a variety of kernels and kernel based classification methods. Finally, the last section is devoted to conclusions and some remarks pertaining to future work.

II. FEATURE EXTRACTION METHODS FOR PHONETIC CLASSIFICATION

Phonetic classification (PC) is concerned with the prediction of the phoneme class given a set of appropriate features that are extracted from the signal. In (PC), it is

assumed that the phoneme boundaries have been already determined by some method. It is considered as a first step towards speech recognition. The most commonly used features for ASR is the Mel Frequency Cepstral Coefficients (MFCC) [8]. They are extracted on a frame by frame basis with fixed overlap between successive windows and characterize the spectral envelope in a short-time frame (typically 10ms) of speech. Despite their popularity, they suffer from many drawbacks such as their failure to capture the spectral frame itself as well as the inherent dynamic information.

Many attempts and references therein, have been made to overcome MFCC's shortcomings and promising alternative frameworks have been introduced for phonetic classification. [9] was the first to suggest the inclusion of MFCC derivatives in an attempt to capture the dynamic information. [3] proposed wavelet filter bank framework for phonetic classification and the authors reported 23.9 % error rate on TIMIT development set. [4] proposed linear discriminate analysis to automatically extract the dynamic information from neighboring static features and reported 29.41% error rate on the complete 39 phonemes test set of TIMIT corpus. [7] Proposed a segmental modeling framework for CDHMM and the authors reported 32.60% error rate on the same corpus. [1] Proposed nonlinear component analysis for computing a statistically independent set of features for CDHMM and an error rate of 26.3% was achieved on the same corpus, outperforming that obtained by MFCC by 2%. Recently, [10] suggest a completely different approach based on recent progress in computational visual neuroscience. Although the error rate obtained for TIMIT vowels is somehow worse than the one obtained using conventional MFCC and the same RLSC algorithm, this approach may be improved and will compete with the MFCC paradigm.

III. SUPPORT VECTOR MACHINES

The main idea of binary SVM is to implicitly map data to a higher dimensional space via a kernel function and then solve an optimization problem to identify the maximum-margin hyper-plane that separates training instances [11]. The separator is based on a set of boundary training examples. Kernels can be interpreted as dissimilarity measures of pairs of objects in the training set X . In standard SVM formulations, the optimal hypothesis sought is of the form (1).

$$\varphi(x) = \sum \alpha_i k(x, x_i) \qquad (1)$$

Where α_i are the components of the unique solution of a linearly constrained quadratic programming problem, whose size is equal to the number of training patterns. The solution vector obtained is generally sparse and the non-zero α_i's are called support vectors (SV's). Clearly, the number of SV's determines the query time which is the time it takes to predict novel observations and subsequently, is critical for some real time applications such as speech recognition tasks. It is worth noting that in contrast to connectionist methods such as neural networks, the examples need not have a Euclidean or fixed-length representation when used in kernel methods.

The training process is implicitly performed in a reproducing space in which $k(x, x_i)$ is the inner product of the images of two examples x, y. Moreover, the optimal hypothesis can be expressed in terms of the kernel that can be defined for non-Euclidean data such biological sequences, speech utterances etc. Popular positive kernels include the Polynomial (2) and the Gaussian (3), kernels:

$$k(x_i, x_j) = (\gamma x_i^T x_j + c)^d, \ \gamma > 0 \qquad (2)$$

$$k(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \gamma > 0 \qquad (3)$$

A). SVM Formulation

Given training vectors $x_i \in \Re^n, i = 1, .., m$, in two classes, and a vector $y \in \Re^m$ such that $y_i \in \{1, -1\}$, support vector classifiers [12, 11, 13] solve the following linearly constrained convex quadratic programming problem:

$$\text{maximize} \quad W(\alpha) \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$

$$\text{subject to} \quad \begin{matrix} \forall i \ 0 \le \alpha_i \le C \\ \sum_{i=1}^{m} \alpha_i y_i = 0 \end{matrix} \qquad (4)$$

The optimal hypothesis is:

$$f(x) = \sum_{i=1}^{m} \alpha_i y_i k(x, x_i) + b \qquad (5)$$

Clearly, the hypothesis f depends only on the non-null coefficients α_i which correspond to the Support Vectors (SVs). The quadratic programming (QP) objective function involves the problem Gram matrix k whose entries are the similarities $k(x, x_i)$ between the patterns x and x_i.

B). SVM Multiclass Extension

Many real applications consist of multiclass classification problems. Unfortunately, SVM is intrinsically bi-class and its efficient extension to multiclass problems is still an ongoing research issue. Several frameworks have been introduced to extend SVM to multiclass contexts and a detailed account of the literature is out of the scope of this paper.

The most common way to build a multiclass SVM is to combine several sub-problems that involve only binary classification. This idea is used by various approaches as One-Versus-All (OVA), One-Versus-One (OVO) [11] and Directed Acyclic Graph (DAGSVM) [14].

In OVA strategy, K binary classifiers are constructed, where K is the number of classes. The classifier is trained by labeling all the examples in the class as positive and the remainder as negative. The final hypothesis is given by the formula:

$$f_{OVA}(x) = \text{argmax}_{i=1..K}\left(f_i(x)\right) \qquad (6)$$

In the second strategy, OVO proceeds by training $\frac{K(K-1)}{2}$ binary classifiers corresponding to all the pairs of classes. The hypothesis consists of choosing either the class with

most votes (voting). In Directed Acyclic Graph (DAGSVM), each node of graph represents a binary classifier (DAGSVM).

IV. RELATED WORK OF FEATURE EXTRACTION

SVM using standard kernel cannot deal directly with variable length or sequential data such as speech patterns. Early implementations attempted to incorporate dynamic information by a hybridization with HMM [15]. In [16], a novel kernel based on Fisher score was introduced and the authors report some positive results. An interesting implementation of SVM for speech patterns which performs frame wise classification was studied in [17]. It is worth while mentioning here that this approach has the advantage of not using phoneme boundaries information and at the same time it can be implemented with standard kernels. However, the size of the training set produced by this method is huge and the authors were forced to use only a portion of the data set for training. They estimated six years of CPU training time for the full TIMIT set. [18] implemented SVM for phonetic classification by using a 3-4-3 rule for producing a fixed-length feature vector from the MFCCs.

The authors report an unusually high recognition rate which we were not able to reproduce. Finally, [19] used linear RLSC for the classification of TIMIT phonemes.

V. REGULARIZATION OF FEATURE EXTRACTION

In this work, we propose three approaches to calculate an adaptive MFCC features for kernel Gaussian methods. Our contribution is based on fragmentation step in MFCC process.

In general, standard MFCC are extracted for 25 milliseconds Hamming windows and 10 milliseconds overlap. The feature vector obtained is of dimension 12 plus an energy term. In this work, the TIMIT datasets contains over 1 million examples, are used to perform the frame wise classification.

Initially, we fix the number of frame (or window) at five; five is the mean of all signals or vectors. We have three approaches:

A). First approach

In this approach, we fix the window (frame) size, and we find the overlap adequate for each phoneme, in order to obtain same, the size for all training and test datasets. For this approach, we fixed the window size for 25 and 30 milliseconds and determinate the overlap by the formula 7, in term of windows step, as shown in the Fig. 1.

In order to keep the size of the training set tractable for kernel methods and take into account the speech dynamics, a natural approach would be to keep the window size fixed and set the window step according to the duration of the phoneme. The window step (overlap) length is computed as:

$$WindowsStep = (Length(Signal) - WindowsSize)/nF \quad (7)$$

Where nF stands for the average number of frames. In our experiments nF was set to 5 resulting in feature vectors of dimension 65 and no derivatives were added.

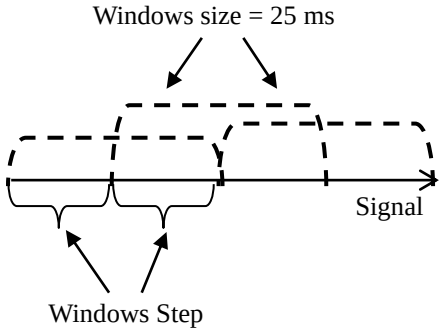

Fig. 1: Illustration of the first approach

B). Second approach

In this second approach, we fix the overlap between two consecutives windows (or frames) in the aim to keep continuous of each signal i.e., regular concatenation of windows, size, and we find the window size adequate for each phoneme, in order to obtain same the size for all training and test datasets, in our case 65 feature.

An overlap of 10 and 12 milliseconds is selected for this approach and the formula 8 is used to calculate the window size, as shown in Fig. 2.

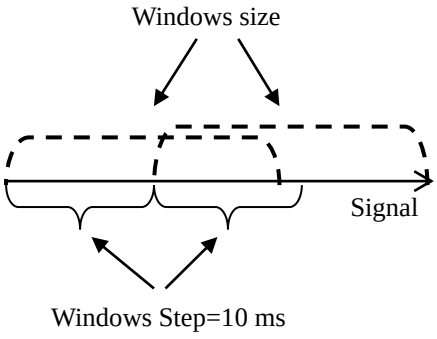

Fig. 2: Illustration of the second approach

The window size length is computed as follow:

$$\text{WindowsSize} = \text{Length(Signal)} - \text{Overlap} * \text{nF} \quad (8)$$

Where *nF* stands for the average number of frames. In our experiments *nF* was set to 5 resulting in feature vectors of dimension 65 and no derivatives were added.

C). Third approach

This approach is mixed approach of the two first approaches, in the aim to minimize the big overlap between two consecutives windows (or frames) and minimize the redundancy of date in windows, i.e., adequate segmentation for each signal phoneme, in order to obtain same the size for all training and test datasets, as shown in Fig. 3.

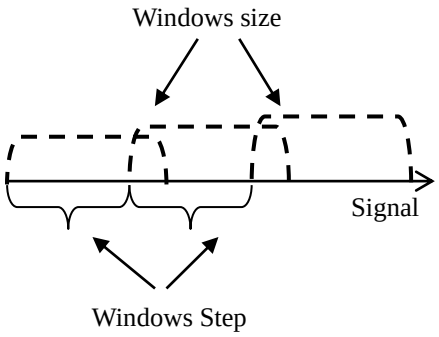

Fig. 3: Illustration of the third approach

V. IMPLEMENTATION AND EXPERIMENTAL RESULTS

In our experiments with the normalization of features extraction framework, or the variable window step feature extraction and variable window size feature extraction, we performed our experiments on the TIMIT corpus. The labels of 20 vowels of **[**19**]** is used in our experiments, as follow : {[aa], [aw], [ae], [ah], [ao], [ax], [ay], [axr], [ax-h], [uw], [ux], [uh], [oy], [ow], [ix], [ih], [iy], [eh], [ey], [er]}. The features extraction or MFCC coefficients are calculated by Auditory Toolbox version 2 [20].

In all the experiments reported below, we performed 3-fold cross validation for tuning SVM hyper parameters g and C. The GNU LibSVM [21] software was used for SVM, with one versus one (OVO) and One versus All (OVA) multi-classes strategies. All the experiments were run on standard Intel i3 3.40 GHZ with 12 Go of memory running the Windows 7 operating system. The following tables summarize our results.

Table I – Table III summarize the best results obtained in the experimentation phase. The parameters of SVM used in our work are: *T*; kernel type; takes the flowing values, Gaussian (2), *C*; regularization parameter of SVM, *g*; Gaussian kernel parameter. The *C* and *g* were calculated by cross-validation. Time (*s*) is time of prediction or test, error(\%) is the error rate of prediction. The data sets are experimented for the OVA and OVO multi-class SVM.

The test rate of prediction is the accuracy, it's used as a statistical measure of how well a classification test correctly identifies. Accuracy is the proportion of true results (both true positives and true negatives) among the total number of cases examined, that is:

$$\text{Accuracy} = \frac{\text{true positives+true negatives}}{\text{total number}} \qquad (9)$$

It is clear that the proposed approaches had given an important improvement on recognition rate. As shown in Fig. 4, the third approach gave the better results, for CPU time all approaches give similar results, as shown in Fig. 5. Finally, we believe that the results obtained are encouraging step to improve the performance of the phonetics recognition.

Table I. Approach one results for 20 phonemes

Window size (ms)	Testing rate (%)	CPU Time (s)	# Test patterns	# SV
30	57.54	101.30	15981	37450
30	58.76	84.55	15981	32475
30	58.71	96.38	15981	32433
30	58.55	92.31	15981	32467
25	57.71	98.71	16395	38366
25	58.60	113.68	16395	32733
25	58.79	106.89	16395	33634
25	58.56	96.27	16395	32839

Test patterns: number of examples used in the test.
SV : number of support vectors.
All experiences is realized with OVA and OVO multiclass SVM.

Table II. Approach two results for 20 phonemes

Window step (ms)	Testing rate (%)	CPU Time (s)	# Test patterns	# SV
10	57.87	101.31	16080	37304
10	58.71	118..27	16080	32597
10	58.73	85.63	16080	32543
10	58.87	89.61	16080	31916
12	57.21	87.39	15345	35625
12	58.52	77.31	15345	30498
12	58.29	81.27	15345	30564
12	58.32	83.46	15345	30589

Test patterns: number of examples used in the test.
SV : number of support vectors.
All experiences are realized with OVA and OVO multiclass SVM.

Table III. Approach three results for 20 phonemes

multiclass SVM	Testing rate (%)	CPU Time (s)	# Test patterns	# SV
OVO	61.17	101.31	16207	32108
OVO	60.91	85.80	16207	32414
OVO	60.96	115.94	16492	32586
OVO	60.94	89.76	16492	32716
OVO	60.76	97.88	16492	31787
OVA	59.71	117.34	16207	37387
OVA	60.20	95.92	16207	36723

Test patterns: number of examples used in the test.
SV : number of support vectors.

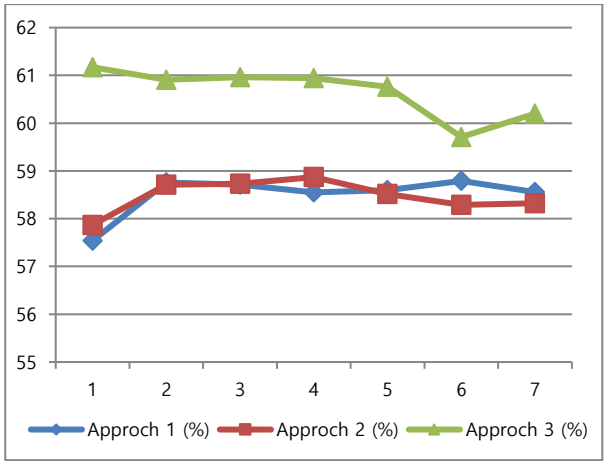

Fig. 4: Variation of error rate for different approaches

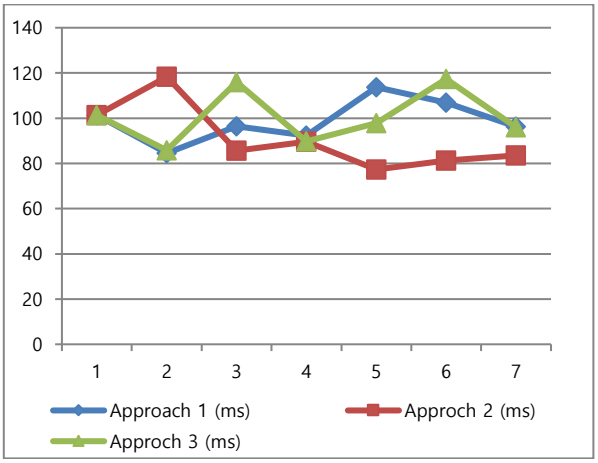

Fig. 5: Variation of recognition time for different approaches

VI. CONCLUSION

We introduced and implemented efficient approaches for phonetics feature extraction. In otherwise, we propose a simple normalization for vocal signals, in aim to obtain vectors or features of identical size, in order to apply classification with SVM machine. The variable window step and window size paradigms were shown through extensive experiments to achieve state of the art results in terms of accuracy. However, as can be seen from the results, testing rate and testing times are still excessive preventing SVM from being used in speech recognition at least for the time being. We believe that in order to improve automatic speech recognition technology, more research efforts should be invested in developing general purpose efficient machine learning paradigms capable of handling large scale multi-class problems.

REFERENCES

[1]. Mohamed Kamal Omar and Mark Hasegawa-Johnson "Non-Linear Independent Component Analysis for Speech Recognition"; *ICCCCT Proceeding, Orlando*, 2003.
[2]. Naomi Harte, Saeed Vaseghi and Paul McCourt. "A Novel Model for Phoneme Recognition Using Phonetically Derived features"; *Proceeding EUSIPCO 1998*.
[3]. F-Ghinwa Choueiter and R-James Glass. "A Wavelet and Filter Bank Framework for Phonetic Classification"; *ICASSP 2005*.
[4]. Hakan Erdogan. "Regularizing Linear Discriminant Analysis for Speech Recognition". 2005.
[5]. Alex Grave and Jürgen Schmidhuber. "Framewise Phoneme Classification with Bidirectional LSTM Networks". *IJCNN 2005*.
[6]. Jesper Salomon, Simon King and Miles Osborne. "Framewise Phone classification Using Support Vector Machines"; *ICSLP 2002*.
[7]. Fei Sha and Lawrence K. Saul "Large Margin Hidden Markov Models for Automatic speech recognition" *NIPS 2006*.
[8]. L. R. Rabiner and B-H Juang. "Fundamentals of speech recognition", Prentice Hall, 1993.
[9]. S. Furui. "Speaker-Independent Isolated Word Recognition Using Dynamic Features of Speech spectrum"; *IEEE Trans, Acoustic, Speech, and Signal Processing* 34, 52-59.
[10]. R. P. Lippmann, "Speech recognition by machines and humans", Speech Communication 22 1-15, Elsevier, 1997.
[11]. Vapnik V, "Statistical learning theory", 736. A Wiley Interscience publication, New York, USA, 1998.
[12]. Vapnik V, "The nature of statistical learning theory", 314. Springer, New York, USA, 2010.
[13]. Vapnik V, Chervonenkis A, "Theory of pattern recognition". USSR, Nauka, Russian, 1974.
[14]. Platt, J. C., Cristianini, N. and Shawe-Taylor, J.2000, Large margin dags for multiclass classification, *in Advances in Neural Information Processing Systems* 12, pp. 547–553, MIT Press.
[15]. A. Ganapathiraju. "Support vector machines for speech recognition". PhD thesis, Mississipi State University, USA, 2001.
[16]. N. Smith and M. Gales. "Speech recognition using SVM. *Advances in Neural Information Processing Systems*", 14, MIT Press, 2002.
[17]. Jesper Salomon, Simon King and Miles Osborne. "Framewise Phone classification Using Support Vector Machines"; *ICSLP 2002*.
[18]. Pedro Moreno. "On the use of Support Vector Machines for Phonetic Classification"; In the *proceedings of ICCASP*, 1999.
[19]. R. Rifkin and a1. "Noise Robust Phonetic Classification with Linear Regularized Least Squares and Second Order Featues"; *ICASSP 2007*.
[20]. M. Slaney, Auditory Toolbox version 2. Tech. Report#010, Internal Research Corporation, 1998.
[21]. C-C. Chang and C-J. Lin. "LIBSVM: a library for support vector machines, 2001", Software available at: http://www.csie.ntu.edu.tw/cjlin/libsvm.