A Speech Recognizer based on Multiclass SVMs with HMM-Guided Segmentation

D. Martín-Iglesias, J. Bernal-Chaves, C. Peláez-Moreno,
A. Gallardo-Antolín and F. Díaz-de-María

Signal Theory and Communications Department
EPS-Universidad Carlos III de Madrid
Avda. de la Universidad, 30, 28911-Leganés (Madrid), SPAIN

Abstract

Automatic Speech Recognition (ASR) is essentially a problem of pattern classification, however, the time dimension of the speech signal has prevented to pose ASR as a simple static classification problem. Support Vector Machine (SVM) classifiers could provide an appropriate solution, since they are very well adapted to high-dimensional classification problems. Nevertheless, the use of SVMs for ASR is by no means straightforward, mainly because SVM classifiers require an input of fixed-dimension. In this paper we study the use of a HMM-based segmentation as a mean to get the fixed-dimension input vectors required by SVMs, in a problem of isolated-digit recognition. Different configurations for all the parameters involved have been tested. Also, we deal with the problem of multi-class classification (as SVMs are initially binary classifers), studying two of the most popular approaches: 1-vs-all and 1-vs-1.

1 Introduction

Hidden Markov Models (HMMs) are, undoubtedly, the most employed core technique for Automatic Speech Recognition (ASR). During the last decades, research in HMMs for ASR has brought about significant advances and, consequently, the HMMs are currently accurately tuned for this application. Nevertheless, we are still far from achieving high-performance ASR systems. Some alternative approaches, most of them based on Artificial Neural Networks (ANNs), were proposed during the last decade ([1], [2], [3], [4] and [5] are some examples). Some of them tackled the ASR problem using predictive ANNs, while others proposed hybrid (HMM-ANN) approaches. Nowadays, however, the preponderance of HMMs in practical ASR systems is a fact.

Speech recognition is essentially a problem of pattern classification, but the high dimensionality of the sequences of speech feature vectors has prevented researchers to propose a straightforward classification scheme for ASR. Support Vector Machines (SVMs) are state-of-the-art tools for linear and nonlinear knowledge discovery [6], [7]. Being based on the maximum margin classifier, SVMs are able to outperform classical classifiers in the presence of high dimensional data even when working with nonlinear machines.

Some researchers have already proposed different approaches to speech recognition aiming at taking advantage of this type of classifiers. Among them, [8], [9] and [10] use different approaches to perform the recognition of short duration units, like isolated phoneme or letter classification. In [8], the authors carry out a length adaptation based on the triphone model approach. In [9] and [10], a normalizing kernel is used to achieve the adaptation. Both cases show the superior discrimination ability of SVMs. Moreover, in [9], a hybrid approach based on HMMs has been proposed and tested in a CSR (Continuous Speech Recognition) task.

Nevertheless, the use of SVMs for ASR is by no means straightforward. The main problem is the required dimensional normalization, due to the fact that the usual kernels can only deal with vectors of fixed size. However, speech analysis generates sequences of feature vectors of variable lengths (due to the different durations of the acoustic units and the constant frame rate commonly employed). A possible solution is that showed in [11], where the non-uniform distribution of analysis instants provided by the internal states of an HMM with a fixed number of states and a Viterbi decoder is used for dimensional normalization.

Another difficulty is that speech recognition is a problem of multi-class classification, while in the original formulation, an SVM is a binary classifier. Although some versions of multi-class SVMs have been proposed, they are computationally expensive. A more usual approach to cope with this limitation is combining a number of binary SVMs to construct a multi-class classifier. In this paper we have studied two of the most popular approaches (1-vs-1 and 1-vs-all), testing it in a specific ASR task.

This paper is organized as follows. In next section, we describe the fundamentals of SVMs and we describe the procedures for multiclass implementation. Afterwards, we make a review of the HMM-guided segmentation method to produce input vectors with fixed dimension. Then, in Section 4, we present the experimental framework and the results obtained, explaining the different criterions followed to chose the several parameters of the system. Finally, some conclusions and further work close the paper.

2 SVM fundamentals

2.1 SVM formulation

An SVM is essentially a binary classifier capable of guessing whether an input vector $\mathbf{x}$ belongs to a class $y_1 = +1$ or to a class $y_2 = -1$. The decision is made according to the following expression:

$$g(\mathbf{x}) = \mathbf{w} \cdot \phi(\mathbf{x}) + b, \tag{1}$$

where $\phi(\mathbf{x}) \ : \ \Re^n \mapsto \Re^{n'}$, $(n << n')$, is a nonlinear function which maps the vector $\mathbf{x}$ to a *feature space* with higher dimensionality (possibly infinite) where classes are linearly separable, and $\mathbf{w}$ defines the separating hyper-plane in such a space.

What makes SVMs more effective than other methods based on linear discriminants is the learning criterion, because instead of minimizing only the empirical risk, they also try to minimize the structural risk, being the solution found a compromise between the empirical error and the generalization capability.

The solution is given by the following minimization problem:

$$\min_{\mathbf{w},b,\xi_i} \quad \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{N} \xi_i, \tag{2}$$

$$\text{subject to} \quad y_i(\mathbf{w} \cdot \phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \tag{3}$$

$$\xi_i \geq 0, \text{ for } i = 1, \cdots, N, \tag{4}$$

where $\mathbf{x}_i \in \Re^n, \ i = 1, \ldots, N$ are the training vectors corresponding to the labels $y_i \in \{\pm 1\}$, and the parameter C establishes the compromise between error minimization and generalization capability.

The SVM is usually solved introducing the restrictions in the minimizing function using Lagrange multipliers, leading to the maximization of the Wolfe dual:

$$L_d = \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_{ij} \alpha_i \alpha_j \phi^T(\mathbf{x}_i) \phi(\mathbf{x}_j) \tag{5}$$

with respect to α_i and subject to $\sum_{i=1}^{n} \alpha_i = 0$ and $0 \leq \alpha_i \leq C$. This problem is quadratic and convex, so its convergence to a global minimum is guaranteed using quadratic programming (QP) schemes. The value of $\mathbf{w}$ and b can be recovered from the Lagrange multipliers α_i, that are associated with the first linear restriction in the SVM formulation:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} \alpha_i y_i \phi(\mathbf{x}_i), \\ b &= \sum_{j} \alpha_j y_j \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) + y_i, \ \forall i. \end{aligned} \tag{6}$$

According to (6), only vectors with an associated $\alpha_i \neq 0$ will contribute to determine the weight vector $\mathbf{w}$ and, therefore, the separating boundary, and they receive the name of *support vectors*.

Generally, function $\phi(\mathbf{x})$ is not explicitly known (in fact, in most of the cases its evaluation would be impossible as long as the feature space dimensionality can be infinite). However, we don't actually need to know it, since the only we need to evaluate are the dot products $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ that, by using what has been called the *kernel trick*, can be evaluated using a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$.

By this way, the form that finally adopts an SVM is:

$$g(\mathbf{x}) = \sum_{i=1}^{N} \lambda_i y_i K(\mathbf{x}_i, \mathbf{x}) + b. \tag{7}$$

The most widely used kernel functions are the *gaussian radial basis function*,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\gamma^2}\right), \tag{8}$$

whith an associated feature space of infinite dimensionality, and the polynomial kernel

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i \cdot \mathbf{x}_j)^p, \tag{9}$$

which associated feature space are the polynomials up to grade p.

2.2 Multiclass SVM

Besides of the necessity of using input vectors with fixed-length, there is another important issue that must be solved whenever we work with SVMs for ASR. While in speech recognition we have to make a decision among several classes, support vector machines were originally designed for binary classification and their generalization to the multi-class case is still an on-going research field.

Although some of the proposed approaches make a reformulation of the SVM equations to consider all classes at once, this option is very expensive computationally and, therefore, we haven't consider them in this work.

Another different approach is combining results of several binary classifiers to construct a multi-class classifier. We have experimented with two different versions of this method. The former consists in comparing each class against all the rest (1-vs-all) while in the latter each class is confronted against all the other classes separately (1-vs-1) [12].

In the 1-vs-all method we have to construct k SVMs (with k the number of classes) and in the 1-vs-1 $k(k-1)/2$, but, since in the second approach the number of training vectors for each class is smaller, the necessary computational effort can be ever lower in the latter case, as shown in [13].

For the 1-vs-1 alternative we have used the implementation described in [14], where error correcting codes are used to compare the outputs of the classifiers, and, for the 1-vs-all approach, we obtained the probability-like outputs using the implementation in [15]. Afterwards, the outputs of the binary 1-vs-all classifiers were compared, and the most probable class among the ones showing a positive output was chosen (positive meaning that the binary classifier had selected the 'one' against the 'rest').

3 Feature Extraction and Dimensional Normalization

Since the speech signal is quasi-stationary, speech analysis must be performed on a short-term basis. Typically, the speech signal is divided into a number of overlapping time windows and a speech feature vector is computed to represent each of these frames. The size of the analysis window, w_a, is usually of 20-30 ms. The frame period, T_f, (the time interval between two consecutive analysis windows) is set to a value between 10 and 15 ms. Habitually, $w_a = KT_f$, where K is called the overlapping factor.

With respect to the feature vectors themselves, for each analysis window, twelve Mel-Frequency Cepstral Coefficients (MFCC) are obtained using a mel-scaled filter-bank with 40 channels. Then, the log-energy, the twelve delta-cepstral coefficients and the delta-log energy are appended, making a total vector dimension of 26.

Typically, the values of w_a and T_f are kept constant for every utterance that, on the other hand, exhibits a different time duration. Consequently, the speech analysis generates sequences of feature vectors of variable length. As we have already mentioned, a normalization of these lengths is required to use SVM classifiers.

In a previous work [11], three procedures to perform this dimensional normalization are proposed. Two of them were very straightforward approaches consisting on adjusting either the analysis window size or the frame period to obtain a fixed number of time analysis instants. The third one, more sophisticated, used and HMM-based segmentation to select the time analysis instants. The next subsection describes this last method, that is the one selected for the experiments conducted in this work.

3.1 Non-uniform distribution of analysis instants

An appropriate selection of the time instants at which the speech signal is analysed can presumably improve the classification results.

To determine the appropriate analysis instants, we propose to use the implicit information in the segmentation made by HMM, i.e., to consider those instants at which state transitions occur (very likely related to those at which the changes of the speech spectra happen).

This HMM-guided parameterization procedure consists of two main stages. The first stage is a HMM classifier (a Viterbi decoder) that yields the best sequence of states for each utterance and also provides a set of state boundary time marks. The second stage extracts the speech feature vectors at the time instants previously marked. For the first stage, we have used left-to-right continuous density HMMs with three Gaussian mixtures per state. Each HMM represents a whole-word and consists of Ns states with the topology shown in Figure 1. These models have been trained using only the training set of the speech database and the conventional parameterization module used for the baseline experiments. In particular, the speech parameters con-sists of 12 MFCC, the log-energy, 12 delta-MFCC and the delta-log energy, extracted using a frame period of 10 ms and an analysis Hamming window of 25 ms.

As mentioned before, these acoustic models are used to generate alignments at state level for each utterance in the speech database. In this process, each utterance is compared to each of the HMMs and only the segmentation produced by the acoustic model yielding the best score is saved for the next stage. Note that the obtained seg-mentation may not be correct, even when the utterance is properly recognized by the HMM-based system. Segmentation errors may produce some degradation in the per-formance of the whole system, however, for our task, the results obtained show that the segmentation is accurate enough. Anyway, it is necessary to consider this issue for further research.

In the second stage, the feature vectors are extracted at the time instants derived from the HMM-guided segmentation. In particular, a 25 ms analysis window is sub-sequently located at these time instants. In this way, the number of feature vectors per utterance used as the SVM input turns out to be equal to the number of states (Ns), determined by the HMM topology. In our case, the number of states was fixed to 17 (the same number of states we use for HMM-based recognition).

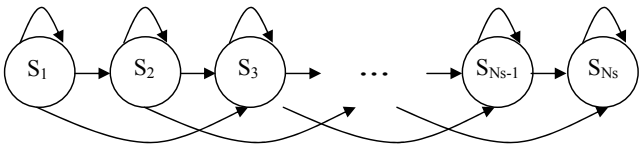

Figure 1: HMM topology

4 Experimental Results

4.1 Baseline System and Database

We have used a database consisting of 72 speakers and 11 utterances per speaker for the 10 Spanish digits. This database was recorded at 8 kHz in clean conditions. Since this database is not large enough to achieve reliable speaker-independent results, we have used a 9-fold cross validation to artificially extend it. Specifically, we have split each database into 9 balanced groups; 8 of them for training and the remaining one for testing, averaging the results afterwards. In summary, we use a total of 7,920 words for testing our systems.

The baseline HMM-based ASR system is an isolated-word, speaker independent system developed using the HTK package [16]. Left-to-right HMMs with continuous observation densities were used. Each of the whole-digit models contains a different number of states (which depends on the number of allophones in the phonetic transcription of each digit) and three Gaussian mixtures per state.

For the baseline experiment with the HMM classifier, a Hamming window with a width of 30 ms was used and the feature vectors (consisting of 12 MFCC, the log-energy, 12 delta-MFCC and the delta-log energy) were extracted once every 10 ms.

We have tested our systems in clean conditions and in presence of additive noise. For that purpose, we have corrupted our database with two kinds of noises, namely: white noise and the noise produced by a F16 plane. Both noises have been extracted from the NOISEX database [17] and added to the speech signal to achieve a signal-to-noise ratio of 12 dB. As we have used clean speech for estimating the acoustic models (in both, HMM and SVM-based recognizers), the noises are only added for testing the recognition performance.

4.2 Selecting parameters for the SVM-based recognizer

In order to get the best performance possible for the system proposed, we have to select properly the values of the different parameters involved. Specifically, we must answer the following questions:

1. What is the best kernel and which are the best parameters for it?
2. How many states must we use in the HMM-segmentation procedure?
3. When must we extract the features, in the transitions between states, or between two transitions, when the voice is stationary?
4. What's the best window size? Does it depend on the length of the utterance?
5. Which parameterization should we use and what is the best normalization procedure?

It is hard to make a guess *a priori* for these questions. However, the answer will have a great impact in the recognition rates reached for our system. For this reason, we ran a set of experiments with different values for all these configuration parameters.

We have used the RBF kernel (eq. (8)) in all the experiments, finding values for γ and for the regularization parameter C of the SVM by using *grid search*. However, we didn't find a significant difference among the different values tried (about 1% in the recognition rate).

Regarding the number of the states of the HMMs used for the segmentation, we have the problem that the different words in the dictionary have different lengths. This implies that for a given number of states, some words can be oversampled while, for others, we won't have taken a number of parameters large enough. For both SVM classifiers (1-vs-all and 1-vs-1), we have finally used a 15 state HMM to produce the sampling instants in which the speech signal is analysed. Thus, in this case we use 15 feature vectors per utterance as the SVM input. The number of 15 was chosen because with less, the recognition rate was poor, and with more, the computational cost was very high, while the improvement in recognition was not so noticeable.

Once selected the most adequate number of states for the HMMs, is necessary to determine the best moment to extract parameters for the SVM. This moment could correspond with the transition between two states, which is associated with a change in the spectrum, or with the time when the voice is stationary, between two transitions. We have tried the two schemes and, even, a combination of both, but all the results were very similar. We finally decided to extract parameters in the transitions between states.

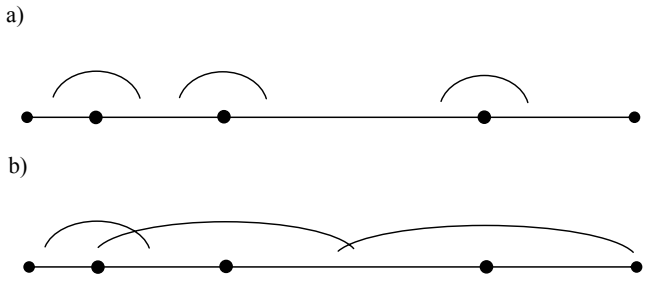

Figure 2: Parameter extraction using a fixed window size a), and a size depending on the length between transitions b).

As long as we are using a fixed number of states for all words, and these words can have very different durations, one could think that it would be a good idea to adjust the size of the window used, making it wider depending on the length between samples (i.e. transitions between states). In Figure 2 is illustrated this approach. In the final implementation, however, as the results with a variable window didn't differ from those obtained with the fixed one, we used the standard fixed window of 30 ms.

	HMM		SVM 1-vs-All		SVM 1-vs-1	
	Clean	12dB WN	Clean	12dB WN	Clean	12dB WN
Baseline	**99.89**	36.92	**99.72**	39.44	**99.72**	37.07
$x_i = x_i - \bar{x}_i$	99.42	41.67	99.51	40.68	99.5	39.87
$x_i = \frac{x_i - \bar{x}_i}{\sigma_{x_i}}$	99.08	31.19	98.8	34.17	98.72	33.29
$x_i = \frac{x_i - \bar{x}_i}{\sigma_x}$	98.32	33.24	98.48	32.23	98.6	31.2
$x_i = \frac{x_i - \bar{x}_i}{\max(x_i)}$	99.34	**49.4**	99.35	**50.73**	99.35	**50.6**

Table 1: Word Accuracy Rate (%) obtained with four different normalizations of the speech features: only subtracting the mean value of each parameter, dividing the result between the standard deviation of each parameter, dividing between the standard deviation of all parameters together and dividing between the maximum value of each parameter.

Concerning parameterization, we used the same as in the baseline experiment (12 MFCC + logE + Δ + Δ-logE). However, an important issue in HMMs, and even more in SVMs, is the normalization used. We tried four different types of normalization, as we can see in Table 1. Although with clean speech the rates without normalization are slightly better, with noisy speech the results obtained subtracting the mean to each parameter and dividing them between their maximum values are the best ones.

4.3 Experiments and Results

On Table 2 are shown the best word recognition rates obtained with both alternatives of the multiclass SVM-based system and in comparison to those achieved by the HMM-based system. As it can be observed, the SVM classifiers performed only slightly better than the baseline system. The explanation for this is that, when HMM fails, the segmentation obtained is far from the optimal and so, the SVMs don't have very much to do. However, even so, SVMs outperforms the HMM-based system in all experiments, getting an improvement of more than 1% in the presence of white noise.

If we compare the two approaches proposed for recognition with SVMs, we can see that results with 1-vs-all performs better than 1-vs-1. However, rates obtained with both methods are again very close.

5 Conclusions and further work

In this paper, we have proposed two different approaches to a multiclass SVM classifier (1-vs-all and 1-vs-1) with application to a specific ASR task. Experimental results have shown that recognition rates obtained with SVM-based systems are very close to that achieved by a conventional HMM-based ASR system in clean conditions. However, in noisy environments, differences are enlarged, getting the 1-vs-all SVM-based classifier the best results.

	Clean	**White (SNR=12 dB)**	**F16 (SNR=12 dB)**
HMM-based ASR	99.34%	49.4%	59.31%
Hybrid HMM-SVM ASR system(1-vs-all)	99.35%	50.73%	59.47%
Hybrid HMM-SVM ASR system(1-vs-1)	99.35%	50.6%	59.42%

Table 2: Recognition results obtained with the two proposed hybrid HMM-SVM-based classifiers for two types of noises (white and F16). Results obtained with the conventional HMM-based ASR system are presented as well.

Although the improvement obtained for the system proposed is not very large, from our point of view, the results are very encouraging since HMM-based systems have been accurately tuned during the last three decades for automatic speech recognition, while speech recognition based on SVMs is a new field of study with a big margin for improvement.

With respect to the further work, we consider several lines: first of all, it would be desirable to find an alternative method of getting a fixed-dimension input, avoiding by this way the problem of a bad segmentation when the HMM fails. Some method based on the behaviour of the derivative of the spectral features could be considered.

Also, since the parameterization used is specially designed for a back-end based on HMMs, it would be interesting to explore alternative parameterizations. Currently, we are completing the first experiments with LSP parameters and the results outperform those showed here.

Finally, we expect to extend the SVM framework for ASR by using string kernels, which has been used with success in tasks as protein [14] and text [17] classification. These kernels, based mainly on the Fisher score and other score spaces, work in conjunction with a generative model, such as an HMM, and can deal with sequences of different length.

References

[1] H. Sakoe, R. Isotani, K. Yoshida, K. Iso, and T. Watanabe. Speaker-independent word recognition using dynamic programming neural networks. *Proc. ICASSP-89*, pages 29–32, 1989.

[2] K. Iso and T. Watanabe. Speaker-independent word recognition using a neural prediction model. *Proc. ICASSP-90*, pages 441–444, 1990.

[3] J. Tebelskis, A. Waibel, B. Petek, and O. Schmidbauer. Continuous speech recognition using predictive neural networks. *Proc. ICASSP-91*, pages 61–64, 1991.

[4] Y. Bengio. Neural networks for speech and sequence recognition. *London International Thomson Computer Press*, 1995.

[5] H.A. Bourlard and N.Morgan. Connectionist speech recognition: a hybrid approach. *Kluwer Academic Publishers*, 1994.

[6] B. Schölkopf and A. Smola. Learning with kernels. *M.I.T. Press*, 2001.

[7] V. Vapnik. Statistical learning theory. *Wiley*, 1998.

[8] P. Clarkson and P.J. Moreno. On the use of support vector machines for phonetic classification. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2:585–588, 1999.

[9] A. Ganapathiraju. Support vector machines for speech recognition. *PhD Thesis, Mississipi State Universisty*, 2002.

[10] N.D. Smith and M.J.F. Gales. Using SVMs and discriminative models for speech recognition. *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2002.

[11] J.M. García-Cabellos, C. Peláez-Moreno, A. Gallardo-Antolín, F. Pérez-Cruz, and F. Díaz de María. SVM classifiers for ASR: A discussion about parameterization. *Proceedings of EUSIPCO 2004*, pages 2067–2070, 2004.

[12] E.L. Allwein, R.E. Schapire, and Y. Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

[13] C.-W. Hsu and C.-J. Lin. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13, 2002.

[14] T.K. Huang, R.C. Weng, and C.J. Lin. A generalized bradley-terry model: From group competition to individual skill. *[on-line]http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/*, 2004.

[15] Ch. Chih-Chung and L. Chih-Jen. LIBSVM: a library for support vector machines. *[on-line]http://www.csie.ntu.edu.tw/ cjlin/libsvm/*, 2004.

[16] S. Young et al. HTK-Hidden Markov Model toolkit (ver 2.1). *Cambridge University*, 1995.

[17] A.P. Varga, J.M. Steenneken, M. Tomlinson, and D. Jones. The NOISEX-92 study on the effect of additive noise on automatic speech recognition. *Tech. Rep. DRA Speech Research Unit*, 1992.