Speech Recognition Based on Discriminative Classifiers

Aldebaro Klautau, Nikola Jevtić and Alon Orlitsky

Abstract— **This work focuses on techniques for using discriminative classifiers such as the support vector machine (SVM) in speech recognition. We review previously proposed architectures and perform experiments with one of them. The results show that using SVMs instead of Gaussian mixtures can bring significant improvements in accuracy, but the computational cost is very high. To circumvent this problem, we propose a new architecture that achieves higher accuracy with a reasonable cost. Preliminary results for a spelling task indicate a decrease of 24% in word error rate over a generative HMM-based system.**

Keywords— **Speech recognition, discriminative training, support vector machines.**

I. Introduction

This work concerns the automatic speech recognition (ASR) problem, which roughly speaking, consists in converting digitized speech into text. More specifically, we study *acoustic modeling*, which together with the *front-end*, *language modeling* and *decoder*, constitute a typical ASR system.

The hidden Markov model (HMM) is the predominant technique in ASR systems [1], [2]. Learning HMMs with maximum likelihood estimation (MLE) corresponds to learning a *generative* sequence classifier [3]. An alternative that often improves accuracy (but increases computational cost) is to adopt *discriminative* learning techniques. Some examples of discriminative learners that have been used in ASR are artificial neural network (ANN) [4], [5], SVM [6], [7] and relevance vector machine (RVM) [8]. Another alternative to the generative HMM system consists in training HMMs in a discriminative way through, e.g., maximum mutual information estimation (MMIE) [9], [10].

In this work, we investigate efficient ways to incorporate discriminative classifiers such SVM [11] to ASR. These classifiers achieved state-of-art results in many pattern recognition applications, but early experiments in ASR have exposed some difficulties. One of them is that the classifier's input in ASR is a variable-length vector. A second issue is that SVM (and similar kernel classifiers) is restricted to binary (two-classes) problems. Another point is that the computational cost of training discriminative classifiers can be prohibitive, given that relatively large datasets are used in ASR. We address all these three issues.

We adopt a hybrid framework in which discriminative classifiers are combined to HMM models, such that the system is able to cope with a variable-length input. We investigate issues with an architecture that has been previously used but has high computational cost. We also propose a new architecture that brings improved accuracy with relatively small computational cost. The proposed architecture allows for adopting distinct features when trying to distinguish different classes of sounds. Hence, we automatically select a *heterogeneous* set of suitable features using a recently proposed algorithm based on *boosting* [12]. The results show that the proposed system outperforms conventional HMM-based ones in preliminary experiments with *continuous speech* (spelling task) using a small corpora.

The work is organized as follows. In Section II we review three architectures for using discriminative classifiers in ASR. Section III discusses one of them in more details, and shows some experimental results. In Section IV we present a new architecture and show preliminary results for continuous speech recognition. Section V presents our final considerations.

II. Architectures for using classifiers in ASR

Different architectures have been recently proposed for using discriminative classifiers in ASR. We start by briefly discussing three of them, which are called here: *phone-based*, *frame-based* and *sequential*.

We want to be able to use classifiers that are restricted to binary problems. However, in ASR the classes can be vowels [13], phones [6] or HMM states [7]. Therefore, all three architectures use an error-correcting output code (ECOC) matrix $\mathbf{M} \in \{-1, 0, 1\}^{Q \times B}$ to decompose the multiclass into binary problems (see, e.g., [14], [15], [16]). The binary classifier f_b is trained according to column $\mathbf{M}(\cdot, b)$. If $\mathbf{M}(q, b) = +1$, all examples of class q are considered *positive*, if $\mathbf{M}(q, b) = -1$, all examples of class q are *negative*, and if $\mathbf{M}(q, b) = 0$, none of the examples of class q participates in the training of classifier f_b.

Two ECOC matrices have been used in ASR: *all-pairs* [7] and *one-vs-rest* [6]. The one-vs-rest matrix induces $B = Q$ binary classifiers $f_1, \ldots, f_Q$, where f_i is trained to distinguish between positive class i and all other negative classes. The all-pairs matrix induces $B = \binom{Q}{2}$ binary classifiers $f_{i,j}, 1 \leq i < j \leq Q$, where $f_{i,j}$ is trained to distinguish between positive class i and negative class j. For example, when $Q = 4$, the one-vs-rest and all-pairs matrices are, respectively,

$$\begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{bmatrix} \text{ and } \begin{bmatrix} +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix}.$$

The authors are with the ECE Department, UCSD, 9500 Gilman Drive, La Jolla, CA 92093, USA. E-mails: a.klautau@ieee.org, nikola@cwc.ucsd.edu and alon@ece.ucsd.edu.

A. Klautau was supported by CAPES, Brazil.

We now point out the specific aspects of each architecture. The frame-based architecture is very similar to the traditional hybrid approach that integrates ANN and HMM [17]. Instead of having an ANN providing the *acoustic scores*, we use binary classifiers combined through an ECOC matrix. Note that the frame-based relies only on binary classifiers, a characteristic that is not shared with the other two architectures.

The phone-based architecture used, e.g., in [18], requires a baseline system (based on HMMs, for example) to perform a first *pass* through the training data. The goal of this first pass is to generate a set of possible segmentations per utterance, where each segmentation indicates the endpoints of the basic units (e.g., phones) associated to the given transcription. These segmentations are organized as N-*best* lists or *lattices* (see, e.g., [2], for definitions), which can then be re-scored by the binary classifiers in a second pass. The key point is that, based on the segmentation provided by the first pass, one can convert each segment (corresponding to a basic unit) into a fixed-length vector, and then use conventional classifiers such as SVM. For example, Ganapathiraju [18] used a linear warping to obtain a fixed-length vector from all frames associated to a phone by the segmentation.

When compared to the frame-based, the phone-based architecture has the advantage of a smaller number N of training examples. According to statistics from the TIMIT corpus [19], American English has an average of 7.2 frames per phone (assuming a frame rate of 100 Hz). Hence, dealing with phones instead of frames can significantly reduce the training time of kernel methods that scale with $\mathcal{O}(N^2)$ or $\mathcal{O}(N^3)$. Another aspect is that, similarly to the situation in segmental modeling [20], segmental features can be used when adopting the phone-based architecture.

An advantage of the frame-based over the phone-based architecture is that the former does not require two passes (so there is no need for using a HMM system in addition), and it is easier to implement in real-time applications. We note that, as done for systems based on segmental modeling [20], it is possible to implement efficient decoders that segment speech on-the-fly (see, e.g., [21]).

The sequential architecture was adopted in [7]. Similar to the phone-based, the sequential architecture uses a baseline HMM system to provide an N-best list. During decoding, only the classifiers associated to states that are active according to the N-best list are used. In [7], the classifiers were SVMs, and they were organized according to an all-pairs ECOC matrix. A strategy equivalent to Hamming decoding (see, e.g, [16]) is adopted, i.e., the SVM scores are quantized and the multiclass decision is based on "voting". The results in terms of word error rate were not as good as frame classification. We note the sequential architecture requires training all the binary classifiers specified by the ECOC matrix, even if only a subset of them will be used for each frame during decoding.

III. Frame-based architecture

Among these three architectures, we chose the frame-based to perform preliminary experiments. Our goal was to investigate some issues related to using SVM in ASR, as discussed below.

A. Design issues

The main problem when adopting the frame-based architecture is the high computational cost during both training and test stages. One simple work-around is to select only a subset of frames when training the system. Unfortunately, our experiments indicate that this strategy implies in a decrease of accuracy [22]. The computational cost of decoding at the test stage depends primarily on the number of support vectors and, as predicted in theory [23], this number increases linearly with N. On the other hand, the accuracy decreases when we unadvisedly throw away frames to use smaller subsets of the training data.

The frame-based architecture requires the specification of a $Q \times B$ ECOC matrix $\mathbf{M}$, where Q and B are the number of shared states and binary classifiers, respectively. One issue is the selection of the number of states per phone. For example, one can keep the same left-right HMM topology with three states, or use only one state per phone. We note that these two alternatives were tested in the ANN / HMM framework, and the latter led to better results [17]. We note that ANN / HMM systems typically use an extended input vector $\mathbf{x}$ obtained by concatenating the features of several (e.g., 9) frames. This way, each input vector $\mathbf{x}$ contains contextual information, which alleviates the fact that only one state is used to model a phone.

Another aspect of the architecture is how to deal with binary learners that are not probabilistic. For example, while RVM returns probabilities that can be readily used, SVM returns real numbers as *scores*. Hence, for non-probabilistic learners, we have to convert the binary scores into probabilities, otherwise the results are very poor.

Finally, we need to convert the B probabilities into an estimate $P(q|\mathbf{x}_t)$ of the posterior probability of state $q = 1, \ldots, Q$ given the input features $\mathbf{x}_t$ of frame t.

Once we have $P(q|\mathbf{x}_t)$, we can use the techniques adopted for hybrid ANN / HMM systems. For example, as explained in [17], the posterior $P(q|\mathbf{x}_t)$ is normalized by the prior $P(q)$, and by Bayes' rule this leads to a *normalized likelihood* $P(q|\mathbf{x}_t)/P(\mathbf{x}_t)$. This likelihood is used to feed the decoder, substituting the likelihoods obtained from mixture of Gaussians in typical HMM-based systems.

As in ANN / HMM systems, the system can be trained using a segmental K-means algorithm [24]. In summary, the segmental K-means uses a pre-existing system to perform, for each utterance, a forced-alignment associating each frame to one state. For training binary classifier f_b, one uses the frame associated to the states q for which $\mathbf{M}(q, b) \neq 0$. After all binary classifiers are trained, they are used to obtain a new forced-alignment and the process is repeated until convergence or a maximum number of iterations is reached.

The next subsection shows experimental results that address the issues involved in the design of an ASR system using the frame-based architecture.

B. Results of classification using SVMs

In this section we show results that are related to the following aspects: training of binary classifiers, conversion of their scores into binary probabilities, estimation of a multiclass probability given these binary probabilities, and comparison of systems based on SVM and Gaussian mixtures.

We used subsets of TIMIT to perform frame and phone classification. The datasets consisted of five highly confusable pairs of phones from TIMIT, belonging to different phonetic classes, such as vowels, nasals, etc. We used context-independent models (without state sharing) and one state per phone.

In all experiments we used SVM with Gaussian kernel as the binary learner, which is well-known for its high accuracy. The SVM kernel parameters were selected through cross-validation (CV) using the training data, as described in [22]. We used a conventional PLP front-end [25], which generates $L = 39$ features (we added estimates of the first two derivatives of 12 PLP coefficients and energy) at a frame rate of 100 Hz. We note that, differently than what is typically done for ANN / HMM, we did not concatenate several frames to include contextual information when composing the input vector $\mathbf{x}$. This way we can directly compare the performance of SVMs and Gaussian mixtures, because the latter are typically used with a single frame as input.

Table I shows the results obtained with the SVMs when considering frame classification, i.e., classifying $\mathbf{x}_t$ for each frame t. We used two different methods to select the kernel parameters. The first consists in using CV on the training set to find the best set of parameters for performing multiclass classification with all 10 phones we are dealing using an all-pairs ECOC matrix. In the second method, we also did CV on the training set, but for each dataset individually, finding the best set of parameters for each binary problem. The results show that the performance is substantially increased when the kernel is tuned for each binary classifier. We note that during the test stage, we want to share kernel computations among binary classifiers to save computations, and the fact that we can have different kernel parameters requires a careful implementation of the decoder.

We now investigate methods to convert the SVM scores into probabilities using phone classification. We used the same set of parameters for all SVMs (column "same" in Table I) and compared two methods: fitting a *sigmoid* and *isotonic regression*, as proposed in [26] and [27], respectively. Table II shows the results. When training the converters, there is a concern with overfitting, given that in the test set, the scores can have different statistics than in the training set. Hence, we compared the conversion methods using the training set to learn the converters and also used 3-fold CV on the training set. The sigmoid outperformed

TABLE I

ERROR RATE FOR SVMS PERFORMING FRAME CLASSIFICATION USING TWO DIFFERENT METHODS TO SELECT THE KERNEL PARAMETERS. THE FIRST USES THE SAME PARAMETERS FOR ALL SVMS, WHILE THE SECOND USES PARAMETERS TUNED TO EACH DATASET INDIVIDUALLY.

	same		individual	
Dataset	train	test	train	test
d-t	3.0	10.8	7.0	10.0
iy-ih	2.4	11.6	4.1	6.5
m-n	3.4	19.1	0.0	14.6
v-f	1.7	4.5	0.8	1.3
z-s	7.2	17.4	4.0	10.2

TABLE III

ERROR RATE (TRAIN / TEST) FOR PHONE CLASSIFICATION USING MIXTURE OF GAUSSIANS AND SVMS.

Dataset	mixture	SVM
d-t	10.5 / 16.0	5.8 / 9.2
iy-ih	7.25 / 9.4	3.8 / 5.4
m-n	11.8 / 17.8	0.0 / 14.4
v-f	4.2 / 3.6	0.5 / 1.1
z-s	13.3 / 15.5	3.2 / 8.2

the isotonic method in this domain, and the numbers indicated that using CV, which increases the computational cost, was not necessary. As expected, we can see that the phone classification results improved upon the associated results for frame classification in Table I.

Table III shows the results comparing SVM and the conventional mixtures of Gaussians. For SVM, we used different kernel parameters per binary problem (column "individual" in Table I) and the sigmoid method (without CV) to convert scores into probabilities. For training the mixtures (i.e., the HMMs), we used the Baum-Welch algorithm. We used 3-state left-right HMMs with 20 Gaussians per mixture. It can be seen that SVM significantly outperforms HMM in this setup of binary phone classification.

Our final experiment was to perform phone classification using the whole TIMIT. We did not use the phone [q] (glottal stop) and collapsed the remaining 60 phones into 39 classes, as conventionally done [28]. The 3-state HMM-based system with 20 Gaussians per mixture achieved a test error of 24.0%.

We used a SVM-based system with the same configuration adopted for obtaining the results in Table III. We used an all-pairs ECOC matrix because it leads to a shorter training time when compared to one-vs-rest. We also limited the number of frames per class to 10 thousand. An extra issue is the conversion of B probabilities from binary classifiers into Q state probabilities. We tested two methods for this task: the *coupling* method[1] proposed in [30] and a method that simply sums the probabilities of all binary classifiers associated to a class and normalize. The coupling method achieved 19.1% of test error, while the second led to 20.2%. We note that, while the HMM system takes less than a day to train using MLE (i.e., Baum-Welch), the SVM-based system takes more than a week.

[1] We note that this method was extended to support an arbitrary ECOC matrix in [29].

TABLE II
Error rate (train / test) for phone classification for different methods of converting scores into probabilities.

Dataset	isotonic (CV)	isotonic	sigmoid (CV)	sigmoid
d-t	1.9 / 10.3	1.8 / 15.0	2.3 / 8.8	1.8 / 7.7
iy-ih	1.8 / 8.5	1.9 / 11.2	1.5 / 8.1	1.5 / 8.5
m-n	1.8 / 14.5	1.6 / 14.3	1.6 / 14.3	1.6 / 14.1
v-f	1.1 / 3.6	0.6 / 6.3	0.9 / 3.1	0.8 / 3.1
z-s	4.4 / 15.3	5.5 / 13.5	4.8 / 13.9	3.9 / 13.7

In summary, the frame-based architecture allows for achieving good results using SVMs, but suffers from an excessive computational cost. In the next section we propose a new architecture, which achieves improvements in accuracy with less computations than the frame-based.

IV. Continuous speech recognition with hot-spot architecture

In this section we propose an architecture for using binary classifiers in ASR that builds upon the sequential architecture. The main motivation is the same: to improve discrimination only when the decoding process is facing problems to distinguish acoustic events. These situations are called here *hot-spots*.

In the *hot-spot* architecture, we use a baseline system composed, for example, by HMMs with shared Gaussian mixtures. We also use a set $\mathcal{H}$ of binary classifiers h that try to detect $H = |\mathcal{H}|$ hot-spots at each frame. Once a hot-spot is detected, the acoustic scores of the baseline system are modified according to the outputs of classifiers $g \in \mathcal{G}$. A hot-spot i is associated to a subset $\mathcal{G}_i$ of $\mathcal{G}$ such that, when detected, the hot-spot fires only a specialized set of classifiers. Alternatively, one can use $\mathcal{G} = \mathcal{H}$, and ask the classifiers $\mathbf{h}$ to do both the job of detecting hot-spots and indicating how the state scores should be modified.

The input vector of a classifier is not restricted to be the same as the input vector $\mathbf{x}$ used by the baseline system. For example, we can use the probabilities[2] $P(\mathbf{x}|q)$ provided by states $q = 1, \ldots, Q$ as additional features. Besides, we allow for a heterogeneous set of features, i.e., each classifier g and h can have its own set of features [12]. We write $h_i(\mathbf{z}^i)$ and $g_j(\mathbf{v}^j)$ in case classifiers h_i and g_j have their own features $\mathbf{z}^i$ and $\mathbf{v}^j$, respectively.

We illustrate the motivation for adopting the hot-spot architecture with an experiment using the TIDIGITS corpus [19] (as typically done, the utterances from kids were excluded). The vocabulary is composed by digits 0-9 and oh, and we adopted a word-loop grammar, i.e., we do not assume any knowledge about the number of digits in an utterance. We used 6-state left-right context-independent HMMs with 8 Gaussians per mixture (the silence model used the same configuration), and 39 features from a conventional MFCC [32] front-end. Using beam-pruning with a relative threshold of 200, allows operation at 0.3 times real-time (on a 1.0 GHz Pentium machine) and the word-error-rate (WER) was 1.2% (147 substitutions, 139 insertions and 54 deletions), with a sentence accuracy of 96.4%.

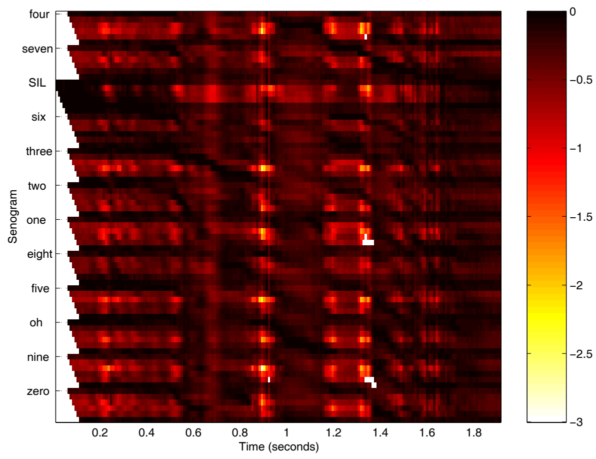


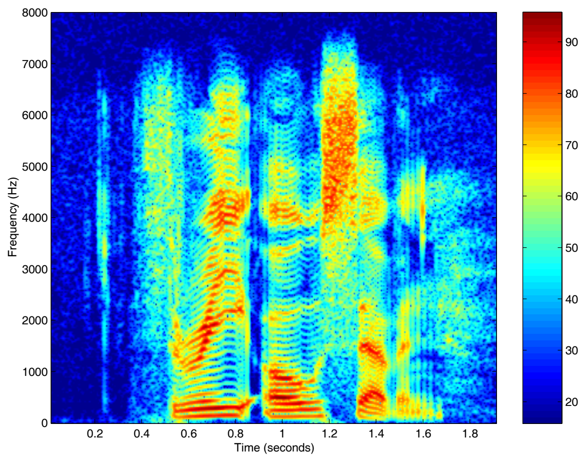
Fig. 1. Senogram for decoding without pruning. The y-axis indicates the HMM model associated to each state, and for each digit, from top to bottom we transverse the 6-state HMM from the leftmost to the rightmost state.

Our architecture tries to exploit the structure of the decoding process. One can visualize such structure by plotting the evolution of the scores along the decoding process. For doing that, we used a representation similar to a spectrogram [2], which was called *senogram*[3] It consists in using a color map to show the evolution of acoustic scores along time (x-axis), ordering the Q states on the y-axis. Figure 1 shows the senogram for an utterance with transcription "three oh seven", and for which the decoder did not use any pruning. It can be seen that all states are active over time. However, this is not what happens in a normal decoding process, where we use pruning to speed-up the search (or even to make it feasible).

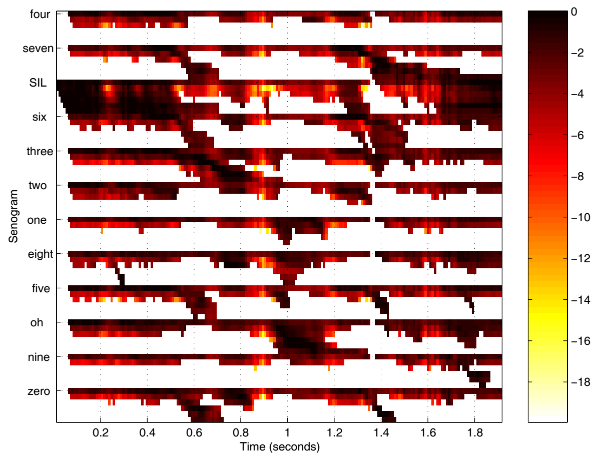
Figure 2 shows a more realistic situation, which corresponds to using the pruning adopted for the baseline system (a threshold of 200). We also show a spectrogram that facilitates tracking the behavior of the system as it decodes the three digits. For example, Figure 2 shows that, when the system is decoding the end of digit "three", the last states of digits "four, two, one, oh and nine" are not *active*. On the other hand, the first state of all models are often active, as the decoder hypothesizes the beginning of a new

[2]Because we use Gaussians, these are not probabilities. Also, to avoid numerical problems, the acoustic score is the log of $P(\mathbf{x}|q)$ in some base, such as 1.0001 [31].

[3]The shared states have been called senones along the development of CMU's Sphinx system [33].



(a)



(b)

Fig. 2. Spectrogram and senogram based on decoding with normal beam-pruning. The utterance is "three oh zero".

word. Of course, the senograms are hard to interpret when one adopts sub-unit (e.g., triphones) models with shared states. Nonetheless, machine learning techniques can still be used to automatically discover the structure imposed by the decoding process.

We now describe our implementation of the hot-spot architecture. For that we need the following definitions. For each utterance $n = 1, \ldots, N$ in the training set, let $\mathcal{T}'_n$ be the correct transcription. Define the *utterance margin* $m(\mathcal{T}) = s(\mathcal{T}') - s(\mathcal{T})$ as the difference between the total scores s of $\mathcal{T}'$ and transcription $\mathcal{T}$. For simplicity, we assume that given a transcription, the score s is obtained with the state alignment that leads to the largest score. Let $\mathcal{T}^* \neq \mathcal{T}'$ be the transcription associated to the hypothesis with largest total score among the ones that survived pruning but do not correspond to $\mathcal{T}'$. Clearly, if $m(\mathcal{T}^*) < 0$, the system incurred in at least one error, and the magnitude of m indicates the level of *confidence* on the decisions. Recall that each utterance is represented by a $L \times T_n$ matrix $\mathbf{X}_n$, where T_n is the number of frames. Hence, for the transcriptions $\mathcal{T}'$ and $\mathcal{T}^*$, the decoder can provide the sequences $\mathbf{q}'$ and $\mathbf{q}^*$, respectively, each with T_n state indices. Similarly, we define the *state margin* $v(t)$ of frame $\mathbf{x}_t$, as the differences in acoustic scores of states q'_t and q^*_t. Note that the state margin is zero if $q'_t = q^*_t$.

We train the system as follows. First we design a baseline using only HMMs (with shared-state triphones in this case). We adopt SVMs as the hot-spot detectors h, and specify their number $|\mathcal{H}|$ beforehand. Each classifier h_i tries to detect difficulty in discriminating between a pair of states (q_i^-, q_i^+), where q_i^- and q_i^+ are associated to "negative" and "positive" labels, respectively. Whenever h_i returns a positive score, another binary classifier g_i (also a SVM) is used to indicate whether the example is negative or positive. Note that in this case $|\mathcal{G}| = |\mathcal{H}|$.

For choosing (q_i^-, q_i^+), we generate lattices using the HMM baseline system and keep only the N_h utterances with smallest utterance margin. We go over these N_h utterances, accumulating the state margin for all $\binom{Q}{2}$ pairs. The $|\mathcal{H}|$ pairs with smallest accumulated state margin correspond to the hot-spots. In our simulations, we selected appropriate features for each classifier using the method proposed in [12]. Besides the 39 MFCC features used by the baseline, we generated 39 PLP features and let the acoustic scores of all states be used as features as well. For example, the digits recognition system with 6-state HMMs used to plot senograms, would lead to a pool of $2 \times 39 + 12 \times 6 = 150$ features to choose from.

To train h_i, we select frames (at most 10000) from the list with N_h utterances as "positive" examples, and frames are randomly picked until we have the same number of "negative" examples.

We tested some alternatives for combining the original acoustic scores of states q_i^- and q_i^+ (provided by the two Gaussians mixtures) with the score provided by the SVM g_i. One method that is computationally simple and worked well consists in converting the score g_i into the range $[0, 1]$ using a sigmoid S_i, then calculate $r = S_i(g_i(\mathbf{z}_t)) - 0.5$ and add νr and $-\nu r$ to the acoustic scores of q_i^+ and q_i^-, respectively, where ν is an empirical constant chosen through experiments with a validation set.

We validated the hot-spot architecture with the following experiment. As baseline system we used the HMM models provided by CMU [33], which were trained using the Wall Street Journal (WSJ) corpus [19]. These models are cross-word context-dependent triphones, with a total of 4147 shared states. The front-end for the baseline system uses MFCCs and generates 39 features. For training the SVMs, we used the training part of the TIMIT corpus. The TIMIT utterances were converted to 39 MFCC-based features and decoded using the WSJ models (we used HTK [34] for generating the lattices). We selected 200 hot-spots based on the pairs with smallest state margin. The features for classifiers h and g were selected from a pool obtained by augmenting the baseline features with the ones described in [12] (we did not use the "duration" feature, so we had a total of 759 features) plus 4147 acoustic scores of states for each frame t. The states that are not active at frame t receive the minimum score at t among all active states. The pool has a total number of $759 + 4147 = 4906$ features per frame, from which we selected only 100 per classifier.

For computational reasons, we used only linear SVMs (while non-linear SVMs often lead to better results), which can be converted to perceptrons. We tested the system using 288 utterances from the "spelling" part of the AN4 corpus, which is described in [35] and freely available at [33]. In summary, the task consists in recognizing the 26 alphabet letters. In this task, the baseline HMM system leads to a WER of 18.1%. Our proposed hot-spot architecture decreased the WER to 13.7%, with the additional overhead of computing the outputs of the perceptrons corresponding to 200 linear SVMs h_i every frame, and the output of g_i when hot-spot i is detected. We also use more CPU cycles for calculating the 759 features, instead of only the MFCC-based. If we do not use the 4147 acoustic scores as features, we reduce the training time but the WER increases to 14.5%.

V. Conclusion

This work focus on alternatives to using classifiers such as SVM in speech recognition systems. We review different architectures, pointing out pros and cons. We perform several experiments with the frame-based architecture and show that it outperforms HMMs with Gaussian mixtures. However, its computational cost can be prohibitive for large corpora. To circumvent this problem, we proposed the new hot-spot architecture, which was able to achieve significant improvements over the conventional generative HMM-based systems in our preliminary experiments.

Some future work includes exploring the several degrees of freedom in the the new architecture, evaluating the impact of each. For example, we used only a linear kernel, while a Gaussian kernel often brings higher accuracy according to the SVM literature. We also compared our system using a limited task. Using larger corpora and comparing the results with HMMs trained in a discriminative way using MMIE are among the next steps.

References

[1] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–86, Feb. 1989.

[2] X. Huang, A. Acero, and H.-W. Hon. *Spoken language processing.* Prentice-Hall, 2001.

[3] Y. Rubinstein and T. Hastie. Discriminative vs informative learning. In *Knowledge Discovery and Data Mining*, pages 49–53, 1997.

[4] H. Bourlard and N. Morgan. *Connectionist speech recognition.* Kluwer, 1994.

[5] F. Beaufays, H. Bourlard, H. Franco, and N. Morgan. Neural networks in automatic speech recognition. *in The Handbook of Brain Theory and Neural Networks*, 2000.

[6] A. Ganapathiraju, J. Hamaker, and J. Picone. Support vector machines for speech recognition. In *ICSLP*, 1998.

[7] S. Fine, G. Saon, and R. Gopinath. Digits recognition in a noisy environment via a sequential GMM/SVM system. In *ICASSP*, 2002.

[8] J. Hamaker, J. Picone, and A. Ganapathiraju. A sparse modeling approach to speech recognition based on relevance vector machines. In *ICSLP*, 2002.

[9] L. Bahl, P. Brown, P. de Souza, and R.L. Mercer (1986). Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *ICASSP*, pages 49–52, 1986.

[10] P. Woodland and D. Povey. Large scale discriminative training of hidden Markov models for speech recognition. *Computer Speech and Language*, 16:25–47, 2002.

[11] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[12] A. Klautau. Mining speech: Automatic selection of heterogeneous features using boosting. In *ICASSP*, 2003.

[13] P. Clarkson and P. Moreno. On the use of support vector machines for phonetic classification. In *ICASSP*, pages 585–8 vol.2, 1999.

[14] T. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artif. Intellig. Research*, 2:263–86, 1995.

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

[16] A. Klautau, N. Jevtić, and A. Orlitsky. On nearest-neighbor ECOC with application to all-pairs multiclass SVM. *to appear in Journal of Machine Learning Research*, 2002.

[17] N. Morgan and H. A. Bourlard. Neural networks for statistical recognition of continuous speech. *Proceedings of the IEEE*, 83(5):742–72, 1995.

[18] A. Ganapathiraju. *Support Vector Machines for Speech Recognition.* PhD thesis, Mississippi State University, 2002.

[19] http://www.ldc.upenn.edu.

[20] M. Ostendorf, V. Digalakis, and O. Kimball. From HMM's to segment models: a unified view of stochastic modeling for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4:360–78, 1996.

[21] S. Lee. *Probabilistic Segmentation for Segment-Based Speech Recognition.* PhD thesis, MIT, 1997.

[22] A. Klautau. *Speech Recognition Using Discriminative Classifiers.* PhD thesis, UCSD, 2003.

[23] B. Scholkopf and A. Smola. *Learning with kernels.* MIT Press, 2002.

[24] L. Rabiner and B. Juang. *Fundamentals of speech recognition.* PTR Prentice Hall, Englewood Cliffs, N.J., 1993.

[25] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *Journal of the Acoustical Society of America*, 87(4):1738–52, Apr. 1990.

[26] J. Platt. Probabilities for sv machines. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

[27] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *KDD*, 2002.

[28] K.-F. Lee and H.-W. Hon. Speaker-independent phone recognition using hidden markov models. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(11):1641–8, Nov. 1989.

[29] B. Zadrozny. Reducing multiclass to binary by coupling probability estimates. In *NIPS*, 2001.

[30] T. Hastie and R. Tibshirani. Classification by pairwise coupling. *The Annals of Statistics*, 26(2):451–471, 1998.

[31] Y. Normandin. *Hidden Markov Models, Maximum Mutual Information Estimation and the Speech Recognition Problem.* PhD thesis, McGill University, 1991.

[32] S. Davis and P. Merlmestein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. on ASSP*, pages 357–366, Aug. 1980.

[33] http://www.speech.cs.cmu.edu/sphinx/.

[34] http://htk.eng.ac.uk.

[35] A. Acero. *Acoustical and environmental robustness in automatic speech recognition.* Kluwer Academic Publishers, Boston, 1993.