# Discriminative Kernel-Based Phoneme Sequence Recognition

*Joseph Keshet*[1], *Shai Shalev-Shwartz*[1], *Samy Bengio*[2], *Yoram Singer*[1,3], *Dan Chazan*[4]

[1]School of Computer Science & Engineering, The Hebrew University, Jerusalem, Israel
[2]IDIAP Research Institute, Martigny, Switzerland
[3]Google Inc., Mountain View, CA
[4]Dept. of Electrical Engineering, Technion, Haifa, Israel

jkeshet@cs.huji.ac.il

## Abstract

We describe a new method for phoneme sequence recognition given a speech utterance, which is not based on the HMM. In contrast to HMM-based approaches, our method uses a discriminative kernel-based training procedure in which the learning process is tailored to the goal of minimizing the Levenshtein distance between the predicted phoneme sequence and the correct sequence. The phoneme sequence predictor is devised by mapping the speech utterance along with a proposed phoneme sequence to a vector-space endowed with an inner-product that is realized by a Mercer kernel. Building on large margin techniques for predicting whole sequences, we are able to devise a learning algorithm which distills to separating the correct phoneme sequence from all other sequences. We describe an iterative algorithm for learning the phoneme sequence recognizer and further describe an efficient implementation of it. We present initial encouraging experimental results with the TIMIT and compare the proposed method to an HMM-based approach.

**Index Terms**: speech recognition, phoneme recognition, acoustic modeling, support vector machines.

## 1. Introduction

Most previous work on phoneme sequence recognition has focused on Hidden Markov Models (HMM). See for example [1, 2, 3] and the references therein. Despite their popularity, HMM-based approaches have several drawbacks such as convergence of the EM procedure to local maximum and overfitting effects due to the large number of parameters. Moreover, HMMs do not faithfully reflect the underlying structure of speech signals as they assume conditional independence of observations given the state sequence [4] and often require uncorrelated acoustic features [5]. Another problem with HMMs is that they do not directly address discriminative tasks. In particular, for the task of phoneme sequence prediction, HMMs as well as other generative models, are not trained to minimize the Levenshtein distance between the model-based predicted phoneme sequence and the correct one.

In this paper we propose an alternative approach for phoneme sequence recognition that builds upon recent work on discriminative supervised learning and overcome the inherent problems of the HMM approaches. The advantage of discriminative learning algorithms stems from the fact that the objective function used during the learning phase is tightly coupled with the decision task one needs to perform. In addition, there is both theoretical and empirical evidence that discriminative learning algorithms are likely to outperform generative models for the same task (see for instance [6, 7]). One of the main goals of this work is to extend the notion of discriminative learning to the complex task of phoneme sequence prediction.

Our proposed method is based on recent advances in kernel machines and large margin classifiers for sequences [8, 9], which in turn build on the pioneering work of Vapnik and colleagues [6, 7]. The phoneme sequence recognizer we devise is based on mapping the speech signal along with the target phoneme sequence into a vector-space endowed with an inner-product that is defined by a kernel operator. One of the well-known discriminative learning algorithms is the support vector machine (SVM), which has already been successfully applied in speech applications [10, 11]. Building on techniques used for learning SVMs, our phoneme sequence recognizer distills to a classifier in this vector-space which is aimed at separating correct phoneme sequences from incorrect ones. The classical SVM algorithm is designed for simple decision tasks such as binary classification and regression. Hence, its exploitation in speech systems so far has also been restricted to simple decision tasks such as phoneme classification. The phoneme sequence recognition problem is more complex, since we need to predict a whole sequence rather than a single number. Previous kernel machine methods for sequence prediction [12, 8, 13] introduce optimization problems which require long run-time and high memory resources, and are thus problematic for the large datasets that are typically encountered in speech processing. We propose an alternative approach which uses an efficient iterative algorithm for learning a discriminative phoneme sequence predictor by traversing the training set a single time.

This paper is organized as follows. In Sec. 2 we formally introduce the phoneme sequence recognition problem. Next, our specific learning method is described in Sec. 3. Our method is based on non-linear phoneme recognition function using Mercer kernels. A specific kernel for our task is presented in Sec. 4. We present preliminary experimental results in Sec. 5 and conclude with a discussion in Sec. 6.

## 2. Problem Setting

In the problem of phoneme sequence recognition, we are given a speech utterance and our goal is to predict the phoneme sequence corresponding to it. We represent a speech signal as a sequence of acoustic feature-vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathbb{R}^d$ for all $1 \leq t \leq T$. We denote the domain of the acoustic feature-vectors by $\mathcal{X} \subset \mathbb{R}^d$. Each utterance corresponds to a sequence of phoneme symbols. Formally, we denote each phoneme symbol by $p \in \mathcal{P}$, where $\mathcal{P}$ is a set of phoneme symbols, and we denote the sequence of phoneme symbols by $\bar{p} = (p_1, \ldots, p_K)$. Furthermore, we denote by $s_k \in \mathbb{N}$ the start time of phoneme $p_k$ (in frame units) and we denote by $\bar{s} = (s_1, \ldots, s_K)$ the sequence of all phoneme start-times. Naturally, the length of the speech signal and hence the number of phonemes varies from one utterance to another and thus $T$ and $K$ are not fixed. We denote by $\mathcal{P}^*$ (and similarly $\mathcal{X}^*$ and $\mathbb{N}^*$) the set of all finite-length sequences over $\mathcal{P}$. Our goal is to learn a function $f$ that predicts the correct phoneme sequence given an acoustic sequence. That is, $f$ is a function from $\mathcal{X}^*$ to the set of finite-length sequences over the domain of phoneme symbols, $\mathcal{P}^*$. We also refer to $f$ as a phoneme sequence recognizer or predictor.

The ultimate goal of the phoneme sequence prediction is usually to minimize the Levenshtein distance between the predicted sequence and the correct one. Throughout this paper we denote by $\gamma(\bar{p}, \bar{p}')$ the Levenshtein distance between the predicted phoneme sequence $\bar{p}'$ and the true phoneme sequence $\bar{p}$. In the next section we present an algorithm which directly aims at minimizing the Levenshtein distance between the predicted phoneme sequence and the correct phoneme sequence.

## 3. The Learning Algorithm

In this section we describe a discriminative supervised learning algorithm for learning a phoneme sequence recognizer $f$ from a training set of examples. Each example in the training set is composed of an acoustic signal $\bar{\mathbf{x}}$, a sequence of phonemes, $\bar{p}$, and a sequence of phoneme start-times, $\bar{s}$.

Our construction is based on a predefined vector feature function $\boldsymbol{\phi} : \mathcal{X}^\star \times (\mathcal{P} \times \mathbb{N})^* \to \mathcal{H}$, where $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS). Thus, the input of this function is an acoustic representation, $\bar{\mathbf{x}}$, together with a candidate phoneme symbol sequence $\bar{p}$ and a candidate phoneme start time sequence $\bar{s}$. The feature function returns a vector in $\mathcal{H}$, where, intuitively, each element of the vector represents the confidence in the suggested phoneme sequence. For example, one element of the feature function can sum the number of times phoneme $p$ comes after phoneme $p'$, while other element of the feature function may extract properties of each acoustic feature vector $\mathbf{x}_t$ provided that phoneme $p$ was pronounced at time $t$. The description of the concrete form of the feature function is differed to Sec. 4.

Our goal is to learn a phoneme sequence recognizer $f$, which takes as input a sequence of acoustic features $\bar{\mathbf{x}}$ and returns a sequence of phoneme symbols $\bar{p}$. The form of the function $f$ we use is

$$f(\bar{\mathbf{x}}) = \arg\max_{\bar{p}} \left( \max_{\bar{s}} \ \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) \right) , \tag{1}$$

where $\mathbf{w} \in \mathcal{H}$ is a vector of importance weights that should be learned. In words, $f$ returns a suggestion for a phoneme sequence by maximizing a weighted sum of the scores returned by the feature function elements. Learning the weight vector $\mathbf{w}$ is analogous to the estimation of the parameters of the local probability functions in HMMs. Our approach, however, does not require $\mathbf{w}$ to take a probabilistic form. The maximization defined by Eq. (1) is over an exponentially large number of all possible phoneme sequences. Nevertheless, as in HMMs, if the feature function, $\boldsymbol{\phi}$, is decomposable, the optimization in Eq. (1) can be efficiently calculated using a dynamic programming procedure.

We now describe a simple iterative algorithm for learning the weight vector $\mathbf{w}$. The algorithm receives as input a training set $S = \{(\bar{\mathbf{x}}_1, \bar{p}_1, \bar{s}_1), \ldots, (\bar{\mathbf{x}}_m, \bar{p}_m, \bar{s}_m)\}$ of examples. Initially we set $\mathbf{w} = \mathbf{0}$. At each iteration the algorithm updates $\mathbf{w}$ according to the $i$th example in $S$ as we now describe. Denote by $\mathbf{w}_{i-1}$ the value of the weight vector before the $i$th iteration. Let $(\bar{p}_i', \bar{s}_i')$ be the predicted phoneme sequence for the $i$th example according to $\mathbf{w}_{i-1}$,

$$(\bar{p}_i', \bar{s}_i') = \arg\max_{(\bar{p}, \bar{s})} \ \mathbf{w}_{i-1} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{p}, \bar{s}) . \tag{2}$$

We set the next weight vector $\mathbf{w}_i$ to be the minimizer of the following optimization problem,

$$\begin{aligned} &\min_{\mathbf{w} \in \mathcal{H}, \xi \geq 0} \ \frac{1}{2} \|\mathbf{w} - \mathbf{w}_{i-1}\|^2 + C\xi \\ &\text{s.t. } \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{p}_i, \bar{s}_i) - \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{p}_i', \bar{s}_i') \geq \gamma(\bar{p}_i, \bar{p}_i') - \xi , \end{aligned} \tag{3}$$

where $C$ serves as a complexity-accuracy trade-off parameter as in the SVM algorithm (see [7]) and $\xi$ is a non-negative slack variable, which indicates the loss of the $i$th example. Intuitively, we would like to minimize the loss of the current example, i.e., the slack variable $\xi$, while keeping the weight vector $\mathbf{w}$ as close as possible to our previous weight vector $\mathbf{w}_{i-1}$. The constraint makes the projection of the correct phoneme sequence $(\bar{\mathbf{x}}_i, \bar{p}_i, \bar{s}_i)$ onto $\mathbf{w}$ higher than the projection of the predicted phoneme sequence $(\bar{p}_i', \bar{s}_i')$ onto $\mathbf{w}$ by at least the Levenshtein distance between them. It can be shown (see [14]) that the solution to the above optimization problem is

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \alpha_i \Delta\boldsymbol{\phi}_i , \tag{4}$$

where $\Delta\boldsymbol{\phi}_i = \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{p}_i, \bar{s}_i) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i, \bar{p}_i', \bar{s}_i')$. The value of the scalar $\alpha_i$ is based on the Levenshtein distance $\gamma(\bar{p}_i, \bar{p}_i')$, the different scores that $\bar{p}_i$ and $\bar{p}_i'$ received according to $\mathbf{w}_{i-1}$, and a parameter $C$. Formally,

$$\alpha_i = \min \left\{ C \ , \ \frac{\max\{\gamma(\bar{p}_i, \bar{p}_i') - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i, 0\}}{\|\Delta\boldsymbol{\phi}_i\|^2} \right\} . \tag{5}$$

The optimization problem given in Eq. (3) is based on ongoing work on online learning algorithms appearing in [14, 9, 15]. These papers demonstrated that, under some mild technical conditions, the cumulative Levenshtein distance of the iterative procedure, $\sum_{i=1}^m \gamma(\bar{p}_i, \bar{p}_i')$, is likely to be small. Moreover, it can be shown [16] that if the cumulative Levenshtein distance of the iterative procedure is small, there exists at least one weight vector among the vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_m\}$ which attains small averaged Levenshtein distance on unseen examples as well. To find this weight vector we simply calculate the averaged Levenshtein distance attained by each of the weight vectors on a validation set.

To conclude this section, we extend the family of linear phoneme sequence recognizers given in Eq. (1) to non-linear recognition functions. This extension is based on Mercer kernels often used in SVM algorithms [6]. Recall that the update rule of the algorithm is $\mathbf{w}_i = \mathbf{w}_{i-1} + \alpha_i \Delta\boldsymbol{\phi}_i$ and that the initial weight vector is $\mathbf{w}_0 = \mathbf{0}$. Thus, $\mathbf{w}_i$ can be rewritten as, $\mathbf{w}_i = \sum_{j=1}^i \alpha_j \Delta\boldsymbol{\phi}_j$ and $f$ can be rewritten as

$$f(\bar{\mathbf{x}}) = \arg\max_{\bar{p}} \ \max_{\bar{s}} \ \sum_{j=1}^{i} \alpha_j \Big( \Delta\boldsymbol{\phi}_j \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) \Big) . \tag{6}$$

By substituting the definition of $\Delta\boldsymbol{\phi}_j$ and replacing the inner-product in Eq. (6) with a general kernel operator $\mathcal{K}(\cdot, \cdot)$ that satisfies Mercer's conditions [6], we obtain a non-linear phoneme recognition function,

$$f(\bar{\mathbf{x}}) = \arg\max_{\bar{p}} \max_{\bar{s}} \sum_{j=1}^{i} \alpha_j \Big( \mathcal{K}(\bar{\mathbf{x}}_i, \bar{p}_i, \bar{s}_i; \bar{\mathbf{x}}, \bar{p}, \bar{s}) - \mathcal{K}(\bar{\mathbf{x}}_i, \bar{p}'_i, \bar{s}'_i; \bar{\mathbf{x}}, \bar{p}, \bar{s}) \Big) . \quad (7)$$

It is easy to verify that the definition of $\alpha_i$ given in Eq. (5) can also be rewritten using the kernel operator.

## 4. Non-Linear Feature Function

In this section we describe the specific feature function we used. As mentioned in the previous section, our goal is to design a non-linear phoneme sequence recognizer using Mercer kernels. Therefore, rather than describing the feature function $\boldsymbol{\phi}$, we describe a kernel operator, which computes implicitly the inner-product $\boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}', \bar{p}', \bar{s}')$. To simplify our notation we denote by $\bar{\mathbf{z}}$ the triplet $(\bar{\mathbf{x}}, \bar{p}, \bar{s})$ and similarly $\bar{\mathbf{z}}'$ denotes $(\bar{\mathbf{x}}', \bar{p}', \bar{s}')$. The kernel operators $\mathcal{K}(\bar{\mathbf{z}}, \bar{\mathbf{z}}')$ we devise can be written as a weighted sum of three kernel operators $\mathcal{K}(\bar{\mathbf{z}}, \bar{\mathbf{z}}') = \sum_{i=1}^{3} \beta_i \mathcal{K}_i(\bar{\mathbf{z}}, \bar{\mathbf{z}}')$, where $\beta_i$ are positive parameters. In the following we describe the three kernel operators we use.

The first kernel operator, $\mathcal{K}_1$, is reminiscent of the acoustic model, which appears in HMMs. First, for each phoneme $p \in \mathcal{P}$, let $T_p(\bar{\mathbf{z}})$ be the set of all frame times in which the phoneme $p$ is uttered. That is, $T_p(\bar{\mathbf{z}}) = \{t : \exists k, \, p_k = p \wedge s_k \le t \le s_{k+1}\}$. Using this definition, the first kernel operator is defined to be

$$\mathcal{K}_1(\bar{\mathbf{z}}, \bar{\mathbf{z}}') \;=\; \sum_{p \in \mathcal{P}} \sum_{t \in T_p(\bar{\mathbf{z}})} \sum_{\tau \in T_p(\bar{\mathbf{z}}')} \exp\left( -\frac{\|\mathbf{x}_t - \mathbf{x}'_\tau\|^2}{2\sigma^2} \right) \;,$$

where $\sigma$ is a predefined constant. We would like to note in passing that the term $\mathcal{K}_1(\bar{\mathbf{z}}_i, \bar{\mathbf{z}}) - \mathcal{K}_1(\bar{\mathbf{z}}'_i, \bar{\mathbf{z}})$ in Eq. (7) can be rewritten in the following more compact form

$$\sum_{p \in \mathcal{P}} \Bigg[ \sum_{t \in T_p(\bar{\mathbf{z}}_i)} \sum_{\tau \in T_p(\bar{\mathbf{z}})} \exp\left( -\frac{\|\mathbf{x}_{i,t} - \mathbf{x}_\tau\|^2}{2\sigma^2} \right) - \sum_{t \in T_p(\bar{\mathbf{z}}'_i)} \sum_{\tau \in T_p(\bar{\mathbf{z}})} \exp\left( -\frac{\|\mathbf{x}_{i,t} - \mathbf{x}_\tau\|^2}{2\sigma^2} \right) \Bigg] .$$

Rewriting the term in the big brackets in a more compact form,

$$\sum_{p \in \mathcal{P}} \sum_{\tau \in T_p(\bar{\mathbf{z}})} \sum_{t=1}^{|\bar{\mathbf{x}}_i|} \psi(t; \bar{\mathbf{z}}_i, \bar{\mathbf{z}}'_i) \exp\left( -\frac{\|\mathbf{x}_{i,t} - \mathbf{x}_\tau\|^2}{2\sigma^2} \right) \;, \quad (8)$$

where

$$\psi(t; \bar{\mathbf{z}}_i, \bar{\mathbf{z}}'_i) \;=\; \begin{cases} 1 & t \in T_p(\bar{\mathbf{z}}_i) \;\wedge\; t \notin T_p(\bar{\mathbf{z}}'_i) \\ -1 & t \notin T_p(\bar{\mathbf{z}}_i) \;\wedge\; t \in T_p(\bar{\mathbf{z}}'_i) \\ 0 & \text{otherwise} \end{cases}$$

In particular, for all frames $\mathbf{x}_{i,t}$ such that $\bar{\mathbf{z}}_i$ and $\bar{\mathbf{z}}'_i$ agree on the uttered phoneme, the value of $\psi(t; \bar{\mathbf{z}}_i, \bar{\mathbf{z}}'_i)$ is zero, which means that frame $\mathbf{x}_{i,t}$ does not effect the prediction.

Before describing $\mathcal{K}_2$ and $\mathcal{K}_3$, we give a simple example of the calculation of $\mathcal{K}_1$. Assume that the phoneme sequence $\bar{p}_i$ is `/f aa r/` with the corresponding start-time sequence $\bar{s}_i = (0, 3, 7)$ and the phoneme sequence $\bar{p}'_i$ is `/f ih r/` with the start-time sequence $\bar{s}'_i = (0, 4, 7)$. Expanding and comparing these sequences, we see that frames 0–2 and 7–9 match while frames 3–6 mismatch (see Table 1). Note that the matched frames do not effect Eq. (8) in any way. In contrast, each mismatched frame influences two summands in Eq. (8): one with a plus sign corresponding to the sequence $\bar{p}_i$ (the phoneme `/aa/`) and one with a minus sign corresponding to the sequence $\bar{p}'_i$ (phonemes `/f/` and `/ih/`).

| frame | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| $\bar{p}$ | f | f | f | **aa** | **aa** | **aa** | **aa** | r | r | r |
| $\bar{p}'$ | f | f | f | **f** | **ih** | **ih** | **ih** | r | r | r |

Table 1: Calculation of $\mathcal{K}_1$: only the bold lettered phonemes are taken into account.

The second kernel operator $\mathcal{K}_2$ is reminiscent of a phoneme duration model and is thus oblivious to the speech signal itself and merely examines the duration of each phoneme. Let $\mathcal{D}$ denote a set of predefined thresholds. For each $p \in \mathcal{P}$ and $d \in \mathcal{D}$ let $N_{p,d}(\bar{\mathbf{z}})$ denote the number of times the phoneme $p$ appeared in $\bar{p}$ while its duration was at least $d$, that is, $N_{p,d}(\bar{\mathbf{z}}) = |\{k : p_k = p \wedge (s_{k+1} - s_k) \ge d\}|$. Using this notation, $\mathcal{K}_2$ is defined to be

$$\mathcal{K}_2(\bar{\mathbf{z}}, \bar{\mathbf{z}}') \;=\; \sum_{p \in \mathcal{P}} \sum_{d \in \mathcal{D}} N_{p,d}(\bar{\mathbf{z}}) \, N_{p,d}(\bar{\mathbf{z}}') \;.$$

The last kernel operator $\mathcal{K}_3$ is reminiscent of a phoneme transition model. Let $A(p', p)$ be an estimated transition probability matrix from phoneme $p'$ to phoneme $p$. Additionally, let $\Theta$ be a set of threshold values. For each $\theta \in \Theta$ let $N_\theta(\bar{\mathbf{z}})$ be the number of times we switch from phoneme $p_{k-1}$ to phoneme $p_k$ such that $A(p_{k-1}, p_k)$ is at least $\theta$, that is, $N_\theta(\bar{\mathbf{z}}) = |\{k : A(p_{k-1}, p_k) \ge \theta\}|$. Using this notation, $\mathcal{K}_3$ is defined to be

$$\mathcal{K}_3(\bar{\mathbf{z}}, \bar{\mathbf{z}}') \;=\; \sum_{\theta \in \Theta} N_\theta(\bar{\mathbf{z}}) \, N_\theta(\bar{\mathbf{z}}') \;.$$

We conclude this section with a brief discussion on the practical evaluation of the function $f$. Recall that calculating $f$ requires solving the optimization problem $f(\bar{\mathbf{x}}) = \arg\max_{\bar{p}} \max_{\bar{s}} \sum_{j=1}^{m} \alpha_j \left( \mathcal{K}(\bar{\mathbf{z}}_j, \bar{\mathbf{z}}) - \mathcal{K}(\bar{\mathbf{z}}'_j, \bar{\mathbf{z}}) \right)$. A direct search for the maximizer is not feasible since the number of possible phoneme sequences is exponential in the length of the sequence. Fortunately, the kernel operator we have presented is decomposable and thus the best phoneme sequence can be found in polynomial time using dynamic programming (similarly to the Viterbi procedure often implemented in HMMs [17]).

## 5. Experimental Results

To validate the effectiveness of the proposed approach we performed experiments with the TIMIT corpus. All the experiments described here have followed the same methodology. We divided the training portion of TIMIT (excluding the SA1 and SA2 utterances) into two disjoint parts containing 3600 and 96 utterances. The first part is used as a training set and the second part is used as a validation set. Mel-frequency cepstrum coefficients (MFCC) along with their first and second derivatives were extracted from the speech waveform in a standard way along with cepstral mean subtraction (CMS). Leading and trailing silences from each utterance were removed. The TIMIT original phoneme set of 61 phonemes was mapped to a 39 phoneme set as proposed by [1].

| | Correct | Accuracy | Ins. | Del. | Sub. |
|---|---|---|---|---|---|
| Kernel-based | 60.8 | 54.9 | 5.9 | 10.8 | 28.4 |
| HMM | 62.7 | 59.1 | 3.6 | 10.5 | 26.8 |

Table 2: Phoneme recognition results comparing our kernel-based discriminative algorithm versus HMM.

Performance was evaluated over the TIMIT core test set by calculating the Levenshtein distance between the predicted phoneme sequence and the correct one.

We applied our method as discussed in Sec. 3 and Sec. 4 where $\sigma^2 = 6, C = 80, \beta = \{1, 0.02, 0.005\}, \mathcal{D} = \{5, 10, 15, \ldots, 40\}$ and $\Theta = \{0.1, 0.2, \ldots, 0.9\}$. We compared the results of our method to the HMM approach, where each phoneme was represented by a simple left-to-right HMM of 5 emitting states with 40 diagonal Gaussians. These models were enrolled as follows: first the HMMs were initialized using K-means, and then enrolled independently using EM. The second step, often called embedded training, re-enrolls all the models by relaxing the segmentation constraints using a forced alignment. Minimum values of the variances for each Gaussian were set to 20% of the global variance of the data. All HMM experiments were done using the *Torch* package [18]. All hyper-parameters including number of states, number of Gaussians per state, variance flooring factor, were tuned using the validation set. The overall results are given in Table 2. We report the number of insertions (Ins.), deletions (Del.) and substitutions (Sub.), as calculated by the Levenshtein distance. The Levenshtein distance is defined as the sum of insertions, deletions, and substitutions. Accuracy stands for 100% minus the Levenshtein distance and Correct stands for Accuracy plus insertions. As can be seen, the HMM method outperforms our method in terms of accuracy, mainly due to the high level of insertions of our method, suggesting that a better duration model should be explored. Nevertheless, we believe that the potential of our method is larger than the results reported and we discuss some possible improvements in the next section. Source code of our method can be found in `http://www.cs.huji.ac.il/~jkeshet/`.

# 6. Discussion

To date, the most successful phoneme sequence recognizers have been based on HMMs. In this paper, we propose an alternative learning scheme for phoneme recognition which is based on discriminative supervised learning and Mercer kernels. The work presented in this paper is part of an ongoing research trying to apply discriminative kernel methods to speech processing problems [19, 15]. So far, the experimental results we obtained with our method for the task of phoneme recognition are still inferior to state-of-the-art results obtained by HMMs. However, while there has been extensive continuous effort on using HMMs for phoneme sequence recognition, our method is rather innovative and the choice of features and kernel operators is by no means comprehensive. We intend to utilize the full power of kernel methods for phoneme recognition by experimenting with additional features and kernels for our task.

**Acknowledgements** Part of this research was done while Joseph Keshet was visiting IDIAP Research Institute. The authors are in debt to Johnny Mariéthoz for making the HMM experiments using the *Torch* package. The authors also would like to thank David Grangier for his many suggestions and comments. This research was supported by the European PASCAL Network of Excellence and by German-Israel Foundation (GIF) under grant number 037-8379.

# 7. References

[1] K.-F. Lee and H.-W. Hon, "Speaker independent phone recognition using hidden markov models," *IEEE Trans. Acoustic, Speech and Signal Proc.*, 37(2), pp. 1641–1648, 1989.

[2] V.V. Digalakis, M. Ostendorf, and J.R. Rohlicek, "Fast algorithms for phone classification and recognition using segment-based models," *IEEE Trans. on Signal Proc.*, vol. 40, pp. 2885–2896, 1992.

[3] R. Chengalvarayan and L. Deng, "Speech trajectory discrimination using the minimum classification error learning," *IEEE Trans. Speech and Audio Proc.*, 6(6), pp. 505–515, 1998.

[4] M. Ostendorf, V.V. Digalakis, and O.A. Kimball, "From hmm's to segment models: A unified view to stochastic modeling for speech recognition," *IEEE Trans. Speech and Audio Proc.*, 4(5), pp. 360–378, 1996.

[5] S. Young, "A review of large-vocabulary continuous speech recognition," *IEEE Signal Proc. Mag.*, pp. 45–57, Sept. 1996.

[6] V. N. Vapnik, *Statistical Learning Theory*, Wiley, 1998.

[7] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, 2000.

[8] B. Taskar, C. Guestrin, and D. Koller, "Max-margin markov networks," in *NIPS*, 2003.

[9] S. Shalev-Shwartz, J. Keshet, and Y. Singer, "Learning to align polyphonic music," in *ISMIR*, 2004.

[10] J. Keshet, D. Chazan, and B.-Z. Bobrovsky, "Plosive spotting with margin classifiers," in *Eurospeech*, 2001.

[11] J. Salomon, S. King, and M. Osborne, "Framewise phone classification using support vector machines," in *ICSLP*, 2002.

[12] M. Collins, "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms," in *EMNLP*, 2002.

[13] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces," in *ICML*, 2004.

[14] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive aggressive algorithms," *Journal of Machine Learning Research*, vol. 7, Mar 2006.

[15] J. Keshet, S. Shalev-Shwartz, Y. Singer, and D. Chazan, "Phoneme alignment based on discriminative learning," in *Interspeech*, 2005.

[16] N. Cesa-Bianchi, A. Conconi, and C. Gentile, "On the generalization ability of on-line learning algorithms," *IEEE Transactions on Information Theory*, 50(9), pp. 2050–2057, 2004.

[17] L. Rabiner and B.H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[18] R. Collobert, S. Bengio, and J. Mariéthoz, "Torch: a modular machine learning software library," IDIAP-RR 46, IDIAP, 2002.

[19] O. Dekel, J. Keshet, and Y. Singer, "Online algorithm for hierarchical phoneme classification," in *Proc. of MLMI*. 2004.