IDIAP RESEARCH REPORT

Discriminative Keyword Spotting

Joseph Keshet [1] David Grangier [1]
Samy Bengio [2]

IDIAP–RR 08-31

March 2008

[1] IDIAP Research Institute, Martigny, Switzerland
[2] Google Inc., Mountain View, CA, USA

IDIAP Research Institute **www.idiap.ch**
Av. des Prés Beudin 20 P.O. Box 592 1920 Martigny – Switzerland
Tel: +41 27 721 77 11 Fax: +41 27 721 77 12 Email: info@idiap.ch

IDIAP Research Report 08-31

Discriminative Keyword Spotting

Joseph Keshet David Grangier Samy Bengio

March 2008

Abstract. This paper proposes a new approach for keyword spotting, which is not based on HMMs. Unlike previous approaches, the proposed method employs a discriminative learning procedure, in which the learning phase aims at maximizing the area under the ROC curve, as this quantity is the most common measure to evaluate keyword spotters. The keyword spotter we devise is based on mapping the input acoustic representation of the speech utterance along with the target keyword into a vector space. Building on techniques used for large margin and kernel methods for predicting whole sequences, our keyword spotter distills to a classifier in this vector-space, which separates speech utterances in which the keyword is uttered from speech utterances in which the keyword is not uttered. We describe a simple iterative algorithm for training the keyword spotter and discuss its formal properties. Experiments on read speech with the TIMIT corpus show that our method outperforms the conventional context-independent HMM-based approach. Further experiments using the TIMIT trained model, but tested on both read (HTIMIT, WSJ) and spontaneous speech (OGI-Stories), show that without further training or adaptation to the new corpus our method outperforms the conventional context-independent HMM-based approach.

1 Introduction

Keyword (or word) spotting refers to the detection of any occurrence of a given word in a speech signal. Most previous work on keyword spotting has been based on hidden Markov models (HMMs). See for example [2, 19, 31, 32] and the references therein. Despite their popularity, HMM-based approaches have several known drawbacks such as convergence of the training algorithm (EM) to a local maxima, conditional independence of observations given the state sequence and the fact that the likelihood is dominated by the observation probabilities, often leaving the transition probabilities unused. However, the most acute weakness of HMMs for keyword spotting is that they do not aim at maximizing the detection rate of the keywords.

In this paper we propose an alternative approach for keyword spotting that builds upon recent work on discriminative large margin and kernel methods, trying to overcome some of the inherent problems of the HMM approaches. Our approach solves directly the keyword spotting problem rather than using a large vocabulary speech recognizer (as in [32]), and does not estimate a garbage or background model (as in[31]). The advantage of discriminative large margin and kernel methods 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 large margin and kernel methods are likely to outperform generative models for the same task (see for instance [11, 34]). One of the main goals of this work is to extend the notion of discriminative large margin and kernel methods to the task of keyword spotting.

Our proposed method is based on recent advances in kernel machines and large margin classifiers for sequences [30, 33], which in turn build on the pioneering work of Vapnik and colleagues [11, 34]. The keyword spotter we devise is based on mapping the speech signal along with the target keyword into a vector-space endowed with an inner-product. Our learning procedure distills to a classifier in this vector-space which is aimed at separating the utterances that contain the keyword from those that do not contain it. On this aspect, our approach is hence related to support vector machine (SVM), which has already been successfully applied in speech applications [16, 29]. However, the model proposed in this paper differs significantly from a classical SVM due to the sequential nature of the keyword spotting problem.

This paper is organized as follows. In Sec. 2 we formally introduce the keyword spotting problem. We then present the large margin approach for keyword spotting in Sec. 3. Next, the proposed iterative learning method is described in Sec. 4. In Sec. 5 we describe the efficient evaluation of our keyword spotter and its complexity. Our method is based on non-linear phoneme recognition and segmentation functions. The specific feature functions we use for are presented in Sec. 6. In Sec. 7 we present experimental results. We conclude the paper in Sec. 8.

Related Work. Most work on keyword spotting has been based on HMMs. In these approaches, the detection of the keyword is based on an HMM composed of two sub-models, the *keyword model* and the background or *garbage model*, such as the HMM depicted in Fig. 6. Given a speech sequence, such a model detects the keyword through Viterbi decoding: the keyword is considered as uttered in the sequence if the best path goes through the keyword model. This generic framework encompasses the three main classes of HMM-based keyword spotters, that is *whole-word* modeling, *phonetic-based* approaches and *large-vocabulary-based*

approaches.

Whole-word modeling is one of the earliest approaches using HMM for keyword spotting [24, 27]. In this context, the keyword model is itself an HMM, trained from recorded utterances of the keyword. The garbage model is also an HMM, trained from non-keyword speech data. The training of such a model hence requires several recorded occurrences of the keyword, in order to estimate reliably the keyword model parameters. Unfortunately, in most applications, such data are rarely provided for training, which yields the introduction of phonetic-based word spotters.

In phonetic-based approaches, both the keyword model and the garbage model are built from phonemes (or triphones) sub-models [4, 20, 26]. Basically, the keyword model is a left-right HMM, resulting from the concatenation of the sub-models corresponding to the keyword phoneme sequence. The garbage model is an ergodic HMM, which fully connects all phonetic sub-models. In this case, sub-model training is performed through embedded training from a large set of acoustic sequences labeled phonetically, like for speech recognition HMMs [23]. This approach hence does not require training utterances of the keyword, solving the main limitation of the whole word modeling approach. However, the phonetic-based HMM has another drawback, due to the use of the same sub-models in the keyword model and in the garbage model. In fact, the garbage model can intrinsically model any phoneme sequence, including the keyword itself. This issue is typically addressed by tuning the prior probability of the keyword, or by using a more refined garbage model, e.g. [4, 20]. Another solution can also be to avoid the need for garbage modeling through the computation of the likelihood of the keyword model for any subsequence of the test signal, as proposed in [15].

A further extension of HMM spotter approaches consists of using Large Vocabulary Continuous Speech Recognition (LVCSR) HMMs. This approach can actually be seen as a phonetic-based approach in which the garbage model only allows valid words from the lexicon, except the targeted keyword. This use of additional linguistic constraints is shown to improve the spotting performance [5, 28, 32, 35]. Such an approach however raises practical concerns: one can wonder whether the design of a keyword spotter should require the expensive collection a large amount of labeled data typically needed to train LVCSR systems, as well as the computational cost implied by large vocabulary decoding [20].

Over the last years, significant effort toward discriminative training of HMMs has been proposed as an alternative to likelihood maximization [1, 14, 13]. These training approaches aim at both maximizing the probability of the correct transcription given an acoustic sequence, and minimizing the probability of the incorrect transcriptions given an acoustic sequence. When applied to keyword spotting, none of these approaches closely tie the training objective with a final spotting objective, such as maximizing the area under the Receiver Operating Curve. In our approach, we reach this goal by proposing a discriminative model focusing on an adequate criterion. In this sense, our work significantly differs from discriminative HMM training for speech recognition, as our learning procedure directly focuses on the spotting performance. Furthermore, we do not constrain the underlying model to be probabilistic, which allows a greater freedom in selecting the set of features.

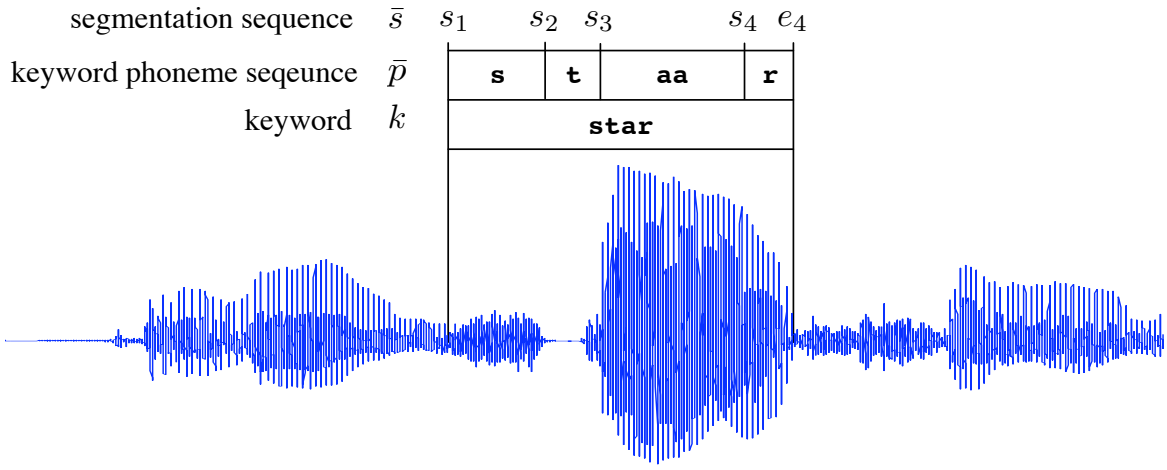

Figure 1: Example of our notation. The waveform of the spoken utterance "a lone star shone..." taken from the TIMIT corpus. The keyword k is the word *star*. The phonetic transcription $\bar{p}$ along with its time span $\bar{s}$ are schematically depicted in the figure.

2 Problem Setting

Any keyword (or word) is naturally composed of a sequence of phonemes. In the keyword spotting task, we are provided with a speech utterance and a keyword and the goal is to identify whether the keyword is uttered at least once in the speech utterance and if so predict its time spans. That is, whether the corresponding sequence of phonemes is articulated in the given utterance and where. We assume that the utterance is small enough for the keyword to be articulated only once. If the utterance is longer then that, we apply the keyword spotter on a sliding window of the appropriate length.

In this section we formally describe the keyword spotting problem. We denote scalars using lower case Latin letters (e.g. x), and vectors using bold face letters (e.g. $\mathbf{x}$). A sequence of elements is designated by a bar ($\bar{\mathbf{x}}$) and its length is denoted as $|\bar{\mathbf{x}}|$.

Formally, we represent a speech signal as a sequence of acoustic feature vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathcal{X} \subset \mathbb{R}^d$ for all $1 \leq t \leq T$. We denote a keyword by $k \in \mathcal{K}$, where $\mathcal{K}$ is a lexicon of words. Each keyword k is composed of a sequence of phonemes $\bar{p}^k = (p_1, \ldots, p_L)$, where $p_l \in \mathcal{P}$ for all $1 \leq l \leq L$ and $\mathcal{P}$ is the domain of the phoneme symbols. We denote by $\mathcal{P}^*$ the set of all finite length sequences over $\mathcal{P}$. Let us further define the alignment between a phoneme sequence and a speech signal. We denote by $s_l \in \mathbb{N}$ the start time of phoneme p_l (in frame units), and by $e_l \in \mathbb{N}$ the end time of phoneme p_l. We assume that the start time of phoneme p_{l+1} is equal to the end time of phoneme p_l, that is, $e_l = s_{l+1}$ for all $1 \leq l \leq L-1$. The timing sequence (time span) $\bar{s}^k$ corresponding to the phonemes sequence $\bar{p}^k$ is a sequence of start-times and an end-time, $\bar{s}^k = (s_1, \ldots, s_L, e_L)$, where s_l is the start-time of phoneme p_l and e_L is the end-time of the last phoneme p_L. An example of our notation is given in Fig. 1. Our goal is to learn a *keyword spotter*, denoted f, which takes as input the pair $(\bar{\mathbf{x}}, \bar{p}^k)$ and returns a real value expressing the confidence that the targeted keyword k is uttered in $\bar{\mathbf{x}}$. That is, f is a function from $\mathcal{X}^* \times \mathcal{P}^*$ to the set

$\mathbb{R}$. The confidence score outputted by f for a given pair $(\bar{\mathbf{x}}, \bar{p}^k)$ can then be compared to a threshold $b \in \mathbb{R}$ to actually determine whether $\bar{p}^k$ is uttered in $\bar{\mathbf{x}}$.

The performance of a keyword spotting system is often measured by the Receiver Operating Characteristics (ROC) curve, that is, a plot of the true positive (spotting a keyword correctly) rate as a function of the false positive (mis-spotting a keyword) rate (see for example [2, 19, 31]). The points on the curve are obtained by sweeping the decision threshold b from the most positive confidence value outputted by the system to the most negative one. Hence, the choice of b represents a trade-off between different operational settings, corresponding to cost functions weighting false positive and false negative errors differently. Assuming a flat prior over all cost functions, it is appropriate to select the keyword spotting system maximizing the averaged performance over all settings, which corresponds to the model maximizing the area under the ROC curve (AUC). In the following we propose an algorithm which directly aims at maximizing the AUC.

3 A Large Margin Approach for Keyword Spotting

In this section we describe a discriminative algorithm for learning a spotting function f from a training set of examples. Our construction is based on a set of predefined feature functions $\{\phi_j\}_{j=1}^n$. Each feature function is of the form $\phi_j : \mathcal{X}^* \times \mathcal{P}^* \times \mathbb{N}^* \to \mathbb{R}$. That is, each feature function takes as input an acoustic representation of a speech utterance $\bar{\mathbf{x}} \in \mathcal{X}^*$, together with a phoneme sequence $\bar{p}^k \in \mathcal{P}^*$ of the keyword k, and a candidate time span $\bar{s}^k \in \mathbb{N}^*$ into an abstract vector-space, and returns a scalar in $\mathbb{R}$ which, intuitively, represents the confidence in the suggested time span given the keyword phoneme sequence $\bar{p}^k$. For example, one feature function can sum the number of times phoneme p comes after phoneme p', while other feature function may extract properties of each acoustic feature vector $\mathbf{x}_t$ provided that phoneme p is pronounced at time t. The description of the concrete form of the feature functions is differed to Sec. 6.

Our goal is to learn a keyword spotter f, which takes as input a sequence of acoustic features $\bar{\mathbf{x}}$, a keyword $\bar{p}^k$, and returns a confidence value in $\mathbb{R}$. The form of the function f we use is

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

where $\mathbf{w} \in \mathbb{R}^n$ is a vector of importance weights ("model parameters") that should be learned and $\boldsymbol{\phi} \in \mathbb{R}^n$ is a vector function composed out of the feature functions ϕ_j. In other words, f returns a confidence prediction about the existence of the keyword in the utterance by maximizing a weighted sum of the scores returned by the feature functions over all possible time spans. The maximization defined by Eq. (1) is over an exponentially large number of time spans. Nevertheless, as in HMMs, if the feature functions $\boldsymbol{\phi}$ are decomposable, the maximization in Eq. (1) can be efficiently calculated through dynamic programming as described in Sec. 5

Recall that we would like to obtain a system that maximizes the AUC on unseen data. In order to do so, we use two sets of training examples. Denote by $\mathcal{X}_k^+$ a set of speech utterances in which the keyword k is uttered. Similarly, denote by $\mathcal{X}_k^-$ a set of speech utterances in which the keyword k is not uttered. The AUC for keyword k can be written in the form of

the *Wilcoxon-Mann-Whitney statistic* [8] as

$$A_k = \frac{1}{|\mathcal{X}_k^+||\mathcal{X}_k^-|} \sum_{\bar{\mathbf{x}}^+ \in \mathcal{X}_k^+} \sum_{\bar{\mathbf{x}}^- \in \mathcal{X}_k^-} \mathbb{1}_{\{f(\bar{\mathbf{x}}^+, \bar{p}^k) > f(\bar{\mathbf{x}}^-, \bar{p}^k)\}}, \tag{2}$$

where $|\cdot|$ refers to the cardinality of a set, and $\mathbb{1}_{\{\cdot\}}$ refers to the indicator function, that is, $\mathbb{1}_{\{\pi\}}$ is 1 whenever the predicate π is true and 0 otherwise. Thus, A_k estimates the probability that the score assigned to an utterance that contains the keyword k is greater than the score assigned to an utterance which does not contain it. Hence, the average AUC over the set of keywords $\mathcal{K}$ can be written as

$$A = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} A_k = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \frac{1}{|\mathcal{X}_k^+||\mathcal{X}_k^-|} \sum_{\bar{\mathbf{x}}^+ \in \mathcal{X}_k^+} \sum_{\bar{\mathbf{x}}^- \in \mathcal{X}_k^-} \mathbb{1}_{\{f(\bar{\mathbf{x}}^+, \bar{p}^k) > f(\bar{\mathbf{x}}^-, \bar{p}^k)\}}. \tag{3}$$

We now describe a large margin approach for learning the weight vector $\mathbf{w}$, which defines the keyword spotting function as in Eq. (1), from a training set S of examples. Each example in the training set S is composed of a keyword phoneme sequence $\bar{p}^k$, an utterance $\bar{\mathbf{x}}^+ \in \mathcal{X}_k^+$ in which the keyword k is uttered, an utterance $\bar{\mathbf{x}}^- \in \mathcal{X}_k^-$ in which the keyword k is not uttered, and a timing sequence $\bar{s}^k$ that corresponds to the location of the keyword in $\bar{\mathbf{x}}^+$. Overall we have m examples, that is, $S = \{(\bar{p}^{k_1}, \bar{\mathbf{x}}_1^+, \bar{\mathbf{x}}_1^-, \bar{s}_1^{k_1}), \ldots, (\bar{p}^{k_m}, \bar{\mathbf{x}}_m^+, \bar{\mathbf{x}}_m^-, \bar{s}_m^{k_m})\}$. Hence, we assume that we have access to the correct start times $\bar{s}^k$ of the phonemes sequence $\bar{p}^k$, for all positive training utterances $\bar{\mathbf{x}}^+ \in \mathcal{X}_k^+$ (and only for these utterances). This assumption is actually not restrictive since such a timing sequence can be inferred by any forced-alignment algorithm [18]. We evaluate the influence of forced-alignment compared to manual-alignment in Sec. 7.

Similarly to the SVM algorithm for binary classification [9, 34], our approach for choosing the weight vector $\mathbf{w}$ is based on the idea of large-margin separation. Theoretically, our approach can be described as a two-step procedure: first, we construct the vectors $\boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}_i^{k_i})$ and $\boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s})$ in the vector space $\mathbb{R}^n$ based on each instance $(\bar{p}^{k_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{s}_i^{k_i})$, and each possible time span $\bar{s}$ for the negative sequence $\bar{\mathbf{x}}_i^-$. Second, we find a vector $\mathbf{w} \in \mathbb{R}^n$, such that the projection of vectors onto $\mathbf{w}$ ranks the constructed vectors according to their quality. Ideally, for any keyword $k_i \in \mathcal{K}_{\text{train}}$, for every instance pair $(\bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-) \in \mathcal{X}_{k_i}^+ \times \mathcal{X}_{k_i}^-$, we would like the following constraint to hold

$$\mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}_i^{k_i}) - \max_{\bar{s}} \ \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s}) \ \geq \ 1 \quad \forall i \ . \tag{4}$$

That is, $\mathbf{w}$ should rank the utterance that contains the keyword above any utterance that does not contain it by at least 1. Moreover, we consider the best possible time span of the keyword within the utterance that does not contain it. We refer to the difference $\mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}_i^{k_i}) - \max_{\bar{s}} \ \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s})$ as the *margin* of $\mathbf{w}$ with respect to the best time span of the keyword k in the utterance that does not contain it. Note that if the prediction of $\mathbf{w}$ is incorrect then the margin is negative. Naturally, if there exists a $\mathbf{w}$ satisfying all the constraints Eq. (4), the margin requirements are also satisfied by multiplying $\mathbf{w}$ by a large scalar. The SVM algorithm solves this problem by selecting the weights $\mathbf{w}$ minimizing $\frac{1}{2}\|\mathbf{w}\|^2$ subject to the constraints given in Eq. (4), as it can be shown that the solution with the smallest norm is likely to achieve better generalization [34].

In practice, it might be the case that the constraints given in Eq. (4) cannot be satisfied. To overcome this obstacle, we follow the soft SVM approach [9, 34] and define the following hinge-loss function,

$$\ell(\mathbf{w};(\bar{p}^k,\bar{\mathbf{x}}^+,\bar{\mathbf{x}}^-,\bar{s}^k)) = \left[1-\mathbf{w}\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}^+,\bar{p}^k,\bar{s}^k)+\max_{\bar{s}}\ \mathbf{w}\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}^-,\bar{p}^k,\bar{s})\right]_+ , \tag{5}$$

where $[a]_+ = \max\{0,a\}$. The hinge loss measures the maximal violation for any of the constraints given in Eq. (4). The soft SVM approach for our problem is to choose the vector $\mathbf{w}^\star$ which minimizes the following optimization problem

$$\mathbf{w}^\star \ = \ \arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\ell(\mathbf{w};(\bar{p}^{k_i},\bar{\mathbf{x}}_i^+,\bar{\mathbf{x}}_i^-,\bar{s}_i^{k_i})) \ \ , \tag{6}$$

where the parameter C serves as a complexity-accuracy trade-off parameter: a low value of C favors a simple model, while a large value of C favors a model which solves all training constraints (see [11]). Solving the optimization problem given in Eq. (6) is expensive since it involves a maximization for each training example. Most of the solvers for this problem, like SMO [22], iterate over the whole dataset several times until convergence. In the next section, we propose a slightly different method, which visits each example only once, and is based on our previous work [10]. Our method is shown to be competitive with the large margin approach and it is shown to maximize the AUC over the training examples and over unseen examples (see Appendix A).

4 An Iterative Algorithm

We now describe a simple iterative algorithm for learning the weight vector $\mathbf{w}$. The algorithm receives as input a set of training examples $S = \{(\bar{p}^{k_i},\bar{\mathbf{x}}_i^+,\bar{\mathbf{x}}_i^-,\bar{s}_i^{k_i})\}_{i=1}^m$ and examines each of them sequentially. Initially, we set $\mathbf{w}=\mathbf{0}$. At each iteration i, the algorithm updates $\mathbf{w}$ according to the current example $(\bar{p}^{k_i},\bar{\mathbf{x}}_i^+,\bar{\mathbf{x}}_i^-,\bar{s}_i^{k_i})$ as we now describe. Denote by $\mathbf{w}_{i-1}$ the value of the weight vector before the ith iteration. Let $\bar{s}'$ be the predicted time span for the negative utterance, $\bar{\mathbf{x}}_i^-$, according to $\mathbf{w}_{i-1}$,

$$\bar{s}' = \arg\max_{\bar{s}}\ \mathbf{w}_{i-1}\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}_i^-,\bar{p}^{k_i},\bar{s}) \ . \tag{7}$$

Let us define the difference between the feature functions of the acoustic sequence in which the keyword is uttered and the feature functions of the acoustic sequence in which the keyword is not uttered as $\Delta\boldsymbol{\phi}_i$, that is,

$$\Delta\boldsymbol{\phi}_i = \boldsymbol{\phi}(\bar{\mathbf{x}}_i^+,\bar{p}^{k_i},\bar{s}^{k_i}) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-,\bar{p}^{k_i},\bar{s}') \ . \tag{8}$$

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

$$\begin{aligned} \min_{\mathbf{w}\in\mathbb{R}^n,\xi\geq 0}\quad & \frac{1}{2}\|\mathbf{w}-\mathbf{w}_{i-1}\|^2 + C\ \xi \\ \text{s.t.}\quad & \mathbf{w}\cdot\Delta\boldsymbol{\phi}_i \geq 1-\xi \ , \end{aligned} \tag{9}$$

where C serves as a complexity-accuracy trade-off parameter (see [10]) and ξ is a non-negative slack variable, which indicates the loss of the ith example. Intuitively, we would like to

Input: training set $S = \{(\bar{p}^{k_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{s}^{k_i})\}_{i=1}^m$; validation set S_{val}; parameter C

Initialize: $\mathbf{w}_0 = \mathbf{0}$

For $i = 1, \ldots, m$

Predict: $\bar{s}' = \arg\max_{\bar{s}} \ \mathbf{w}_{i-1} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s})$

Set: $\Delta\boldsymbol{\phi}_i \ = \ \boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}^{k_i}) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s}')$

If $\mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i < 1$

Set: $\alpha_i = \min\left\{C \ , \ \dfrac{1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i}{\|\Delta\boldsymbol{\phi}_i\|^2}\right\}$

Update: $\mathbf{w}_i = \mathbf{w}_{i-1} + \alpha_i \cdot \Delta\boldsymbol{\phi}_i$

Output: The weight vector $\mathbf{w}^*$ which achieves best AUC performance on the validation set S_{val}.

Figure 2: An iterative algorithm.

minimize the loss of the current example, i.e., the slack variable ξ, while keeping the weight vector $\mathbf{w}$ as close as possible to the previous weight vector $\mathbf{w}_{i-1}$. The constraint makes the projection of the sequence that contains the keyword onto $\mathbf{w}$ higher than the projection of the sequence that does not contain it onto $\mathbf{w}$ by at least 1. It can be shown (see [10]) that the solution to the above optimization problem is

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \alpha_i \Delta\boldsymbol{\phi}_i \ . \tag{10}$$

The value of the scalar α_i is based on the difference $\Delta\Phi_i$, the previous weight vector $\mathbf{w}_{i-1}$, and a parameter C. Formally,

$$\alpha_i = \min\left\{C, \ \frac{[1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i]_+}{\|\Delta\boldsymbol{\phi}_i\|^2}\right\} \ . \tag{11}$$

The optimization problem given in Eq. (9) is based on recent work on online learning algorithms [10]. Based on this work, it is shown in Appendix A that, under some mild technical conditions, the cumulative AUC of the iterative procedure, i.e., $\frac{1}{m}\sum_{i=1}^m \mathbb{1}_{\{\mathbf{w}_i \cdot \Delta\boldsymbol{\phi}_i > 0\}}$ is likely to be high. Moreover, the appendix further shows that given the high cumulative AUC, there exists at least one weight vector among the vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_m\}$ which attains high averaged AUC on unseen examples as well. To find such a weight vector, we simply calculate the averaged loss attained by each of the weight vectors on a validation set. A pseudo-code of our algorithm is given in Fig. 2.

In the case the user would like to select a threshold b that would ensure a specific requirement in terms of true positive rate or false negative rate, a simple cross-validation procedure (see [3]) would consist in selecting the confidence value given by our model at the point of interest over the ROC curve plotted for some validation utterances of the targeted keyword.

5 Efficiency and Complexity

We now describe the problem of efficient evaluation of the function f given in Eq. (1). the evaluation of f requires solving the following optimization problem,

$$f(\bar{\mathbf{x}}, \bar{p}^k) = \max_{\bar{s}} \mathbf{w} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{p}^k, \bar{s}) \quad .$$

Similarly, we need to find an efficient way for solving the maximization problem given in Eq. (7). A direct search for the maximizer is not feasible since the number of possible time spans, $\bar{s}$, is exponential in the number of events. Fortunately, as we show below, by imposing a few mild conditions on the structure of the feature functions both problems can be solved in polynomial time.

For simplicity, we assume that each feature function, ϕ_j, can be decomposed as follows. Let $\hat{\phi}_j$ be any function from $\mathcal{X}^* \times \mathcal{P}^* \times \mathbb{N}^3$ into the reals, which can be computed in a constant time. That is, $\hat{\phi}_j$ receives as input the signal, $\bar{\mathbf{x}}$, the sequence of phonemes, $\bar{p}^k$, and three time points. Additionally, we use the convention $s_0 = 0$ and $s_{|\bar{p}^k|+1} = T + 1$. Using the above notation, we assume that each ϕ_j can be decomposed to be

$$\phi_j(\bar{\mathbf{x}}, \bar{p}^k, \bar{s}) \;=\; \sum_{i=2}^{|\bar{s}|-1} \hat{\phi}_j(\bar{\mathbf{x}}, \bar{p}^k, s_{i-1}, s_i, s_{i+1}) \quad . \tag{12}$$

The feature functions we describe in the next section can be decomposed as in Eq. (12).

We now describe an efficient algorithm for calculating the best time span assuming that ϕ_j can be decomposed as in Eq. (12). Given phoneme index $i \in \{1, \ldots, |\bar{p}^k|\}$ and two time indices $t, t' \in \{1, \ldots, T\}$, denote by $D(i, t, t')$ the score for the prefix of the phoneme index sequence $1, \ldots, i$, assuming that their actual start times are $s_1, \ldots, s_i$, where $s_i = t'$ and assuming that $s_{i+1} = t$. This variable can be computed efficiently in a similar fashion to the forward variables calculated by the Viterbi procedure in HMMs (see for instance [23]). The pseudo code for computing $D(i, t, t')$ recursively is shown in Fig. 3. The best time span, $\bar{s}^\star$, is obtained from the algorithm by saving the intermediate values that maximize each expression in the recursion step. The complexity of the decoding is $\mathcal{O}(|\bar{p}^k|\,|\bar{\mathbf{x}}|^4)$. However, in practice, we can use the assumption that the maximal length of an event is bounded, $t - t' \leq L_{\max}$. This assumption reduces the complexity of the decoding down to $\mathcal{O}(|\bar{p}^k|\,|\bar{\mathbf{x}}|\,L_{\max}^3)$. For comparison, the complexity of the decoding in standard Viterbi-based HMM is $\mathcal{O}(|P + \bar{p}^k|\,|\bar{\mathbf{x}}|)$.

To conclude this section we discuss the global complexity of our proposed method. In the training phase, our algorithm performs m iterations, one iteration per training example. At each iteration the algorithm evaluates the keyword spotting function once, updates the keyword spotting function, if needed, and evaluates the new function on a validation set of size m_{val}. Each evaluation of the function takes an order of $\mathcal{O}(|\bar{p}|\,|\bar{\mathbf{x}}|\,L_{\max}^3)$ operations. Therefore the total complexity of our method becomes $\mathcal{O}(m\; m_{\text{val}}\,|\bar{p}|\,|\bar{\mathbf{x}}|\,L_{\max}^3)$. In practice, however, we can evaluate the updated keyword spotting function only for the last 20 iterations or so, which reduces the global complexity of the algorithm to $\mathcal{O}(m\,|\bar{p}|\,|\bar{\mathbf{x}}|\,L_{\max}^3)$. In all of our experiments, evaluating the keyword spotting function only for the last 20 iterations was found empirically to give sufficient results.

Input: speech signal $\bar{\mathbf{x}}$; sequence of phonemes $\bar{p}^k$; weight vector $\mathbf{w}$; maximum phoneme duration $L_{\max}$; minimum phoneme duration $L_{\min}$

Initialize: $\forall(1 \leq t \leq L_{\max}),\ D(0,t,0) = 0$

Recursion:

For $s = 1,\ldots,T$

For $i = 1,\ldots,|\bar{p}^k|$

For $t = s + iL_{\min},\ldots,s + iL_{\max}$

For $t' = t - L_{\max},\ldots,t - L_{\min}$

$$D(i,t,t') = \max_{t'-L_{\max} \leq t'' < t'-L_{\min}} D(i-1,t',t'') + \mathbf{w} \cdot \hat{\boldsymbol{\phi}}(\bar{\mathbf{x}},\bar{p},t'',t',t)$$

$$D^\star_s = \max_{s+|\bar{p}^k|L_{\min} \leq t \leq s+|\bar{p}^k|L_{\max}} \ \max_{t-L_{\max} \leq t' \leq t-L_{\min}} D(|\bar{p}^k|,t,t')$$

Termination: $D^\star = \max_s D^\star_s, \quad \bar{s}^\star = \arg\max_s D^\star_s$

Figure 3: An efficient procedure for evaluating the keyword spotting function.

6 Feature Functions

In this section we present the implementation details of our learning approach for the task of keyword spotting. Recall that our construction is based on a set of feature functions, $\{\phi_j\}_{j=1}^n$, which maps an acoustic-phonetic representation of a speech utterance as well as a suggested time span of the keyword into a vector-space. In order to make this section more readable we omit the keyword index k.

We introduce a specific set of base functions, which is highly adequate for the keyword spotting problem. We utilize seven different feature functions ($n = 7$). These feature functions are used for defining our keyword spotting function $f(\bar{\mathbf{x}},\bar{p})$ as in Eq. (1). Note that the same set of feature functions is also useful in the task of large-margin forced-alignment [18], and they are given here only for completeness. A detailed analysis of this feature set is given in [18].

Our first four feature functions aim at capturing transitions between phonemes. These feature functions are the distance between frames of the acoustic signal at both sides of phoneme boundaries as suggested by a timing sequence $\bar{s}$. The distance measure we employ, denoted by d, is the Euclidean distance between feature vectors. Our underlying assumption is that if two frames, $\mathbf{x}_t$ and $\mathbf{x}_{t'}$, are derived from the same phoneme then the distance $d(\mathbf{x}_t,\mathbf{x}_{t'})$ should be smaller than if the two frames are derived from different phonemes. Formally, our first four feature functions are defined as

$$\phi_j(\bar{\mathbf{x}},\bar{p},\bar{s}) = \frac{1}{|\bar{p}|}\sum_{i=2}^{|\bar{p}|-1} d(\mathbf{x}_{-j+s_i},\mathbf{x}_{j+s_i}), \quad j \in \{1,2,3,4\}\ . \tag{13}$$

If $\bar{s}$ is the correct time span then distances between frames across the phoneme change points are likely to be large. In contrast, an incorrect phoneme start time sequence is likely to compare frames from the same phoneme, often resulting in small distances.

The fifth feature function we use is built from a frame-wise phoneme classifier described in [12]. Formally, for each phoneme event $p \in \mathcal{P}$ and frame $\mathbf{x} \in \mathcal{X}$, there is a confidence, denoted $g_p(\mathbf{x})$, that the phoneme p is pronounced in the frame $\mathbf{x}$. The resulting feature function measures the cumulative confidence of the complete speech signal given the phoneme sequence and their start-times,

$$\phi_5(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \frac{1}{|\bar{p}|} \sum_{i=1}^{|\bar{p}|} \frac{1}{s_{i+1} - s_i} \sum_{t=s_i}^{s_{i+1}-1} g_{p_i}(\mathbf{x}_t) \ . \tag{14}$$

Our next feature function scores timing sequences based on phoneme durations. Unlike the previous feature functions, the sixth feature function is oblivious to the speech signal itself. It merely examines the length of each phoneme, as suggested by $\bar{s}$, compared to the typical length required to pronounce this phoneme. Formally,

$$\phi_6(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \frac{1}{|\bar{p}|} \sum_{i=1}^{|\bar{p}|} \log \ \mathcal{N}(s_{i+1} - s_i; \hat{\mu}_{p_i} \ , \hat{\sigma}_{p_i}) \ \ , \tag{15}$$

where $\mathcal{N}$ is a Normal probability density function with mean $\hat{\mu}_p$ and standard deviation $\hat{\sigma}_p$. In our experiments, we estimated $\hat{\mu}_p$ and $\hat{\sigma}_p$ from the training set (see Sec. 7).

Our last feature function exploits assumptions on the speaking rate of a speaker. Intuitively, people usually speak in an almost steady rate and therefore a timing sequence in which speech rate is changed abruptly is probably incorrect. Formally, let $\hat{\mu}_p$ be the average length required to pronounce the pth phoneme. We denote by r_i the relative speech rate, $r_i = (s_{i+1} - s_i)/\hat{\mu}_{p_i}$. That is, r_i is the ratio between the actual length of phoneme p_i as suggested by $\bar{s}$ to its average length. The relative speech rate presumably changes slowly over time. In practice the speaking rate ratios often differ from speaker to speaker and within a given utterance. We measure the local change in the speaking rate as $(r_i - r_{i-1})^2$ and we define the feature function ϕ_7 as the local change in the speaking rate,

$$\phi_7(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \frac{1}{|\bar{p}|} \sum_{i=2}^{|\bar{p}|} (r_i - r_{i-1})^2 \ \ . \tag{16}$$

7 Experimental Results

In this section we present experimental results that demonstrate the robustness of our algorithm. We performed experiments on read speech using the TIMIT, HTIMIT and WSJ corpora and on spontaneous speech using the OGI Stories corpus. In all the experiments, the baseline discriminative system and HMM system were trained on the clean read-speech TIMIT corpus. We divided the training portion of TIMIT (excluding the SA1 and SA2 utterances) into two disjoint parts containing 500, and 3196 utterances. The first part of the training set was used for learning the functions g_p (Eq. (14)), which define the feature function ϕ_5. These functions were learned by the algorithm described in [12] using the MFCC+Δ+$\Delta\Delta$ acoustic features and a Gaussian kernel with parameter $\sigma = 6.24$. Although the training of these functions is based on a single parameter σ, the functions are composed of a set of support patterns, which typically include thousands of training examples (in our case there

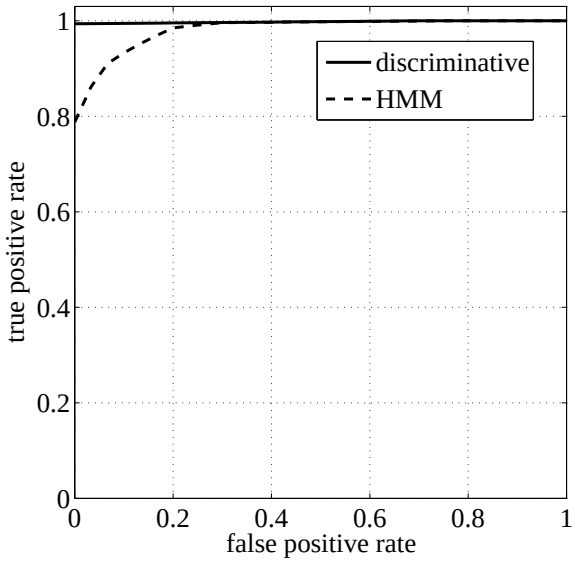

Figure 4: ROC curves of the discriminative algorithm and the HMM approach, trained on the TIMIT training set and tested on 80 keywords from TIMIT test set. The AUC of the ROC curves is **0.995** and **0.941** for the discriminative algorithm and the HMM algorithm, respectively.

were $104,600$ support patterns). Using the functions g_p as a frame-based phoneme classifier resulted in classification accuracy of 55% per frame on the TIMIT test set.

The second set of 3196 utterances formed the training set for the keyword spotter. From this set we picked 200 random keywords for training and 200 different keywords for validation. The keywords were chosen to have a minimum length of at least 6 phonemes. For each of the keywords we chose one positive utterance in which the keyword was pronounced and one negative utterance in which the keyword was not pronounced. The same utterance could be a positive utterance for one keyword and a negative utterance for a different keyword, but in any case the utterances used for the training were not used for validation. The discriminative algorithm was very robust to the set of training and validation keyword set and picking a different set led to similar performance results. We ran the iterative discriminative algorithm with value of the parameter $C = 1$.

Since our method is context-independent, we compared it to context-independent HMM approach. We trained a context-independent HMM phoneme recognizer from the entire TIMIT training portion, where 3600 utterances were used as a training set and 96 utterances were used as a validation set. In our setting 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 and the segmentation provided by TIMIT. 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 [7]. All

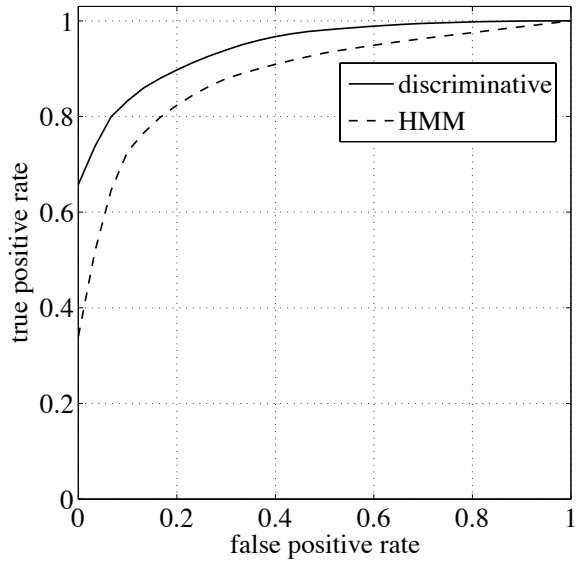

Figure 5: ROC curves of the discriminative algorithm and the HMM approach, trained on the TIMIT training set and tested on 80 keywords from WSJ test set. The AUC of the ROC curves is **0.942** and **0.88** for the discriminative algorithm and the HMM algorithm, respectively.

hyper-parameters including number of states, number of Gaussians per state, variance flooring factor, were tuned using the validation set. The number of parameters in the HMM model can be calculated as follows. There are 5 states per phone, 40 Gaussians per states, 39 phones, the data is 39 dimensional. There are $(40 + 2 \times 40 \times 39) \times 5 \times 39$ parameters for emission and $(8 \times 39) + (39 \times 39)$ parameters for transitions. Overall there are $618,033$ parameters in the HMM system. The resulting HMM was a context-independent state-of-the-art phoneme recognizer with accuracy of 64% on the TIMIT test set.

Keyword detection was performed with a new HMM composed of two sub HMM models, the keyword model and the garbage model, as depicted in Fig. 6. The keyword model was an HMM which estimated the likelihood of an acoustic sequence given that the sequence represented the keyword phoneme sequence. The garbage model was an HMM composed of phoneme HMMs fully connected to each others, which estimated the likelihood of any acoustic sequence. The overall HMM fully connected the keyword model and the garbage model. The detection of a keyword given a test utterance was performed through a best path search, were an external parameter of the prior keyword probability was added to the keyword sub HMM model. The best path found by Viterbi decoding on the overall HMM either passed through the keyword model (in which case the keyword was said to be uttered) or not (in which case the keyword was not in the acoustic sequence). Swiping the prior keyword probability parameters set the trade-off between the true positive rate and the false positive rate.

The test set was composed of 80 randomly chosen keywords, distinct from the keywords of the training and validation sets (the list of keyword for this experiment as well as for all

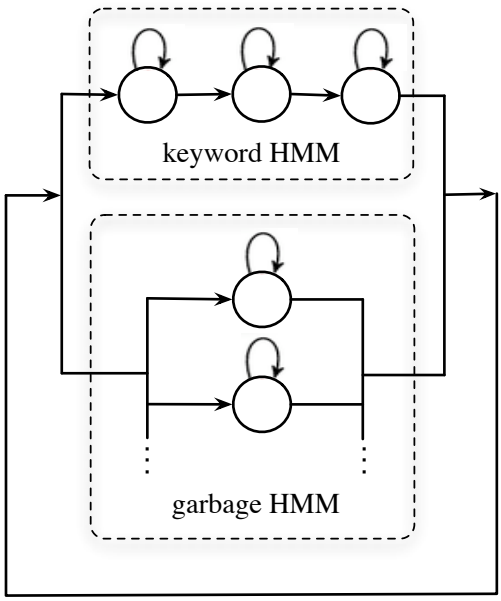

Figure 6: HMM topology for keyword spotting.

other experiments is given in Appendix B). The keywords were selected from the TIMIT dictionary to have a minimal length of 4 phonemes. For each keyword, we randomly picked at most 20 utterances in which the keyword was uttered and at most 20 utterances in which it was not uttered. Note that the number of test utterances in which the keyword was uttered was not always 20, since some keywords were uttered less than 20 times in the whole TIMIT test set. Both the discriminative algorithm and the HMM based algorithm were evaluated against the test data. The results are reported as averaged ROC curves in Fig. 4. The AUC of the ROC curves is 0.995 and 0.941 for the discriminative algorithm and the HMM algorithm, respectively. In order to check whether the advantage over the averaged AUC could be due to a few keyword, we ran the Wilcoxon test. At the 95% confidence level, the test rejected this hypothesis, showing that our model indeed brings a consistent improvement on the keyword set.

In order to make sure that the our learning procedure can be applied in absence of manual alignment data, we trained a model on phoneme time spans extracted from forced aligned data. We used the algorithm presented in [18] for forced alignment, where it was trained on 50 utterances of TIMIT training portion which was not used for training or validation the keyword spotting algorithm. The AUC of the discriminative algorithm trained with forced aligned data was 0.996, almost identical to the AUC of the algorithm trained on manual aligned data. The results are given in Table 1.

In the next experiments we examine the robustness of the proposed algorithm to different environments. We used the model trained on TIMIT but we tested it on different corpora without any further training or adaptation. For the discriminative model, we used the manually aligned trained system, since there was no significant difference between the manually aligned and the forced aligned systems. First we checked the discriminative model on the HTIMIT corpus [25]. The HTIMIT corpus was generated by playing the TIMIT speech

Table 1: Comparison of training the discriminative model with the TIMIT manual phoneme alignment and with automatic forced-alignment. The AUC of the discriminative model on the TIMIT test set is compared to the HMM.

Training alignment	Discriminative Algo. test set AUC	HMM test set AUC
TIMIT manual alignment	0.995	0.941
TIMIT forced alignment	0.996	0.941

through a loudspeaker into a different set of phone handsets. The TIMIT trained systems were tested on a set of 80 keywords which were not used in the training set. For each keyword, we randomly picked at most 20 utterances in which the keyword was uttered and at most 20 utterances in which it was not uttered from the CB1 portion of the HTIMIT corpus. The AUC of the ROC curves was 0.949 and 0.922 for the discriminative algorithm and the HMM algorithm, respectively. With more than 99% confidence, the Wilcoxon test rejected the hypothesis that the difference between the two models was due to only a few keywords. Hence, these experiments on HTIMIT show that the introduction of channel variations degrades the performance of both models, but does not change the relative advantage of our approach over the HMM.

Next, we compared the performance of the proposed discriminative algorithm and of the HMM on the Wall Street Journal (WSJ) corpus [21]. This corpus corresponds to read articles of the Wall Street Journal, and hence presents a different linguistic context compared to TIMIT. Both the discriminative model and the HMM were trained on the TIMIT corpus as described above and tested on a different set of 80 keywords from the WSJ corpus. For each keyword, we randomly picked at most 20 utterances in which the keyword was uttered and at most 20 utterances in which it was not uttered from the `si_tr_s` portion of the WSJ corpus. The ROC curves are given in Fig. 5. The AUC of the ROC curves is 0.942 and 0.88 for the discriminative algorithm and the HMM algorithm, respectively. With more than 99% confidence, the Wilcoxon test rejected the hypothesis that the difference between the two models was due to only a few keywords.

Last, we compared the performance of the algorithms on OGI Stories corpus[1]. In this corpus, spontaneous speech was recorded by asking American speakers to talk freely about a topic of their choice. Again, both systems were trained on the TIMIT corpus as described above and tested on a different set of 60 keywords. For each keyword, we randomly picked at most 20 utterances in which the keyword was uttered and at most 20 utterances in which it was not uttered. The results of these experiments on spontaneous speech are consistent with the results obtained on read speech. Indeed, the discriminative approach outperforms the HMM, with an AUC of 0.769 compared to 0.722 for the HMM. As in previous cases, the Wilcoxon test rejected the hypothesis that the difference between the two models was due to only a few keywords, at the 95% confidence level.

A summary of the results of all experiments is given in Table 2. A closer look on them

[1] `http://cslu.cse.ogi.edu/corpora/stories/`

Table 2: The AUC of the discriminative model compared to the HMM in the experiments.

	Discriminative Algo.	HMM
Corpus	AUC	AUC
TIMIT	**0.996**	0.941
HTIMIT	**0.949**	0.922
WSJ	**0.942**	0.87
OGI Stories	**0.769**	0.722

shows that the discriminative algorithm systematically outperforms the HMM in terms of AUC. This indeed validates our hypothesis that it is a good strategy to maximize the AUC. Moreover, the discriminative algorithm outperforms the HMM for all point of the ROC curve, meaning that it has better true positive rate for every given false negative rate.

8 Summary

Keyword spotting is a speech related task with more and more practical interest from an application point of view. Current state-of-the-art approaches are based on classical generative HMM based systems. In this work, we introduced a discriminative approach to keyword spotting, directly optimizing an objective function related to the area under the ROC curve, i.e., the most common measure for keyword spotter evaluation. Furthermore, the proposed approach is based on a large-margin formulation of the problem (hence expecting a good generalization performance) and an iterative training algorithm (hence expecting to scale reasonably well to large databases). Compared to conventional context-independent HMM approach, the proposed model has shown to yield a statistically significant improvement on the TIMIT corpus. Furthermore, the very same model trained on the TIMIT corpus was tested on different corpora to assess its performance in various conditions. Namely, the model has been assessed on HTIMIT, which introduces various channel variations, on WSJ, which introduces different types of sentences from a linguistic perspective, and on OGI Stories, which corresponds to the recording of spontaneous speech. In all cases, the discriminative algorithm was shown to yield a statistically better performance than the HMM alternative.

We would like to note that this work is part of a general line of research on large margin and kernel method for discriminative continuous speech recognition. [12] described and analyzed an hierarchical approach for frame-based phoneme classification. Building on that work, we proposed an large-margin based discriminative algorithm for forced-alignment [18], and an algorithm for whole sequence phoneme recognition [17]. The discriminative keyword spotting presented in this paper is in turn based on those works and it is the first to address word-level recognition. We are currently investigating an extension of this work to large-margin discriminative large vocabulary continuous speech recognition. We are also looking for a method to encompass contextual information into our models. Last but not least, we are working on reducing the heavy computational load required by kernel-based algorithms, with the objective to reach the efficiency of HMM-based solutions.

Acknowledgements

This work was supported by EU project DIRAC (FP6-0027787).

A Theoretical Analysis

In this appendix, we show that the iterative algorithm given in Sec. 4 maximizes the cumulative AUC, defined as

$$\tilde{A} = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}_{\{\mathbf{w}_i\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}_i^+,\bar{p}^{k_i},\bar{s}^{k_i})\geq\mathbf{w}_i\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}^-,\bar{p}^{k_i},\bar{s}_i')\}}. \tag{17}$$

Our first theorem shows that the area above the curve, i.e. $1-\tilde{A}$, is smaller than the average loss of the solution of the SVM problem defined in Eq. (6). That is, the cumulative AUC, generating by the iterative algorithm is going to be large, given that the loss of the SVM solution (or any other solution) is small, and that the number of examples, m, is sufficiently large.

Theorem 1. *Let $S = \{(\bar{p}^{k_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{s}_i^{k_i})\}_{i=1}^m$ be a set of training examples and assume that for all k, $\bar{\mathbf{x}}$ and $\bar{s}$ we have that $\|\boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{p}^k, \bar{s})\| \leq 1$. Let $\mathbf{w}^\star$ be the optimum of the SVM problem given in Eq. (6). Let $\mathbf{w}_1, \ldots, \mathbf{w}_m$ be the sequence of weight vectors obtained by the algorithm in Fig. 2 given the training set S. Then,*

$$1 - \tilde{A} \leq \frac{1}{m}\|\mathbf{w}^\star\|^2 + \frac{2C}{m}\sum_{i=1}^{m}\ell(\mathbf{w}^\star; (\bar{p}^{k_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{s}^{k_i})). \tag{18}$$

where $C > 1$ and $\tilde{A}$ is the cumulative AUC.

Proof The proof of the theorem relies on Lemma 1 and Theorem 4 in [10]. Lemma 1 in [10] implies that,

$$\sum_{i=1}^{m}\alpha_i\left(2\ell_i - \alpha_i\|\Delta\boldsymbol{\phi}_i\|^2 - 2\ell_i^\star\right) \leq \|\mathbf{w}^\star\|^2. \tag{19}$$

Now if the algorithm makes a prediction mistake, i.e., predicts that an utterance that does not contain the keyword has a greater confidence than another utterance that does contain it, then $\ell_i \geq 1$. Using the assumption that $\|\boldsymbol{\phi}(\bar{\mathbf{x}}, \bar{p}^k, \bar{s})\| \leq 1$ and the definition of α_i given in Eq. (11), when substituting $[1 - \mathbf{w}_{i-1}\cdot\Delta\boldsymbol{\phi}_i]_+$ for ℓ_i in its denominator, we conclude that if a prediction mistake occurs then it holds that

$$\alpha_i\ell_i \geq \min\left\{\frac{\ell_i}{\Delta\boldsymbol{\phi}_i}, C\right\} \geq \min\{1, C\} = 1. \tag{20}$$

Summing over all the prediction mistakes made on the entire training set S and taking into account that $\alpha_i\ell_i$ is always non-negative. it holds that

$$\sum_{i=1}^{m}\alpha_i\ell_i \geq \sum_{i=1}^{m}\mathbb{1}_{\{\mathbf{w}_i\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}_i^+,\bar{p}^{k_i},\bar{s}^{k_i})\leq\mathbf{w}_i\cdot\boldsymbol{\phi}(\bar{\mathbf{x}}^-,\bar{p}^{k_i},\bar{s}_i')\}}. \tag{21}$$

Again using the definition of α_i, we know that $\alpha_i \ell_i^\star \leq C \ell_i^\star$ and that $\alpha_i \|\Delta\boldsymbol{\phi}_i\|^2 \leq \ell_i$. Plugging these two inequalities and Eq. (21) into Eq. (19) we get

$$\sum_{i=1}^{m} \mathbb{1}_{\{\mathbf{w}_i \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}^{k_i}) \leq \mathbf{w}_i \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}^-, \bar{p}^{k_i}, \bar{s}_i')\}} \leq \|\mathbf{w}^\star\|^2 + 2C \sum_{i=1}^{m} \ell_i^\star . \tag{22}$$

The theorem follows by replacing the sum over prediction mistakes to a sum over prediction hits and plugging-in the definition of the cumulative AUC given in Eq. (17). □

The next theorem states that the output of our algorithm is likely to have good generalization, i.e. the expected value of the AUC resulted from decoding on unseen test set is likely to be large.

Theorem 2. *Under the same conditions of Thm. 1. Assume that the training set S and the validation set S_{val} are both sampled i.i.d. from a distribution Q. Denote by m_{val} the size of the validation set. With probability of at least $1-\delta$ we have*

$$1 - \hat{A} = \mathbb{E}_Q \left[\mathbb{1}_{\{f(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) \leq f(\bar{\mathbf{x}}^-, \bar{p}^{k_i})\}} \right] = \Pr_Q \left[f(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) \leq f(\bar{\mathbf{x}}^-, \bar{p}^{k_i}) \right] \leq$$
$$\frac{1}{m} \sum_{i=1}^{m} \ell(\mathbf{w}^\star; (\bar{p}^{k_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{s}^{k_i})) + \frac{\|\mathbf{w}^\star\|^2}{m} + \frac{\sqrt{2\ln(2/\delta)}}{\sqrt{m}} + \frac{\sqrt{2\ln(2m/\delta)}}{\sqrt{m_{\text{val}}}} \quad , \tag{23}$$

where $\hat{A}$ is the mean AUC defined as $\hat{A} = \mathbb{E}_Q \left[\mathbb{1}_{\{f(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) > f(\bar{\mathbf{x}}^-, \bar{p}^{k_i})\}} \right]$.

Proof Denote the risk of keyword spotter f by

$$\text{risk}(f) = \mathbb{E} \left[\mathbb{1}_{\{f(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) \leq f(\bar{\mathbf{x}}^-, \bar{p}^{k_i})\}} \right] = \Pr \left[f(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) \leq f(\bar{\mathbf{x}}^-, \bar{p}^{k_i}) \right]$$

Proposition 1 in [6] implies that with probability of at least $1-\delta_1$ the following bound holds,

$$\frac{1}{m} \sum_{i=1}^{m} \text{risk}(f_i) \leq \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\{f_i(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) \leq f_i(\bar{\mathbf{x}}^-, \bar{p}^{k_i})\}} + \frac{\sqrt{2\ln(1/\delta_1)}}{\sqrt{m}} \quad .$$

Combining this fact with Thm. 1 we obtain that,

$$\frac{1}{m} \sum_{i=1}^{m} \text{risk}(f_i) \leq \frac{2C}{m} \sum_{i=1}^{m} \ell_i^\star + \frac{\|\mathbf{w}^\star\|^2}{m} + \frac{\sqrt{2\ln(1/\delta_1)}}{\sqrt{m}} \quad . \tag{24}$$

The left-hand side of the above inequality upper bounds $\text{risk}(f^\star)$, where $f^\star = \arg\min_{f_i} \text{risk}(f_i)$. Therefore, among the finite set of keyword spotting functions, $F = \{f_1, \ldots, f_m\}$, there exists at least one keyword spotting function (for instance the function $f^\star$) whose true risk is bounded above by the right hand side of Eq. (24). Recall that the output of our algorithm is the keyword spotter $f \in F$, which minimizes the average cost over the validation set S_{val}. Applying Hoeffding inequality together with the union bound over F we conclude that with probability of at least $1-\delta_2$,

$$\text{risk}(f) \leq \text{risk}(f^\star) + \sqrt{\frac{2\ln(m/\delta_2)}{m_{\text{val}}}} \quad ,$$

where $m_{\text{val}} = |S_{\text{val}}|$. We have therefore shown that with probability of at least $1 - \delta_1 - \delta_2$ the following inequality holds,

$$\text{risk}(f) \;\leq\; \frac{1}{m}\sum_{i=1}^{m} \ell_i^\star + \frac{\|\mathbf{w}^\star\|^2}{m} + \frac{\sqrt{2\ln(1/\delta_1)}}{\sqrt{m}} + \frac{\sqrt{2\ln(m/\delta_2)}}{\sqrt{m_{\text{val}}}} \quad .$$

Setting $\delta_1 = \delta_2 = \delta/2$ concludes our proof. □

B Lists of Keywords

We give here the list of keywords used in the experiments described in Sec. 7.

The keywords used in the TIMIT experiments were: absolute, admitted, aligning, anxiety, apartments, apparently, argued, bedrooms, brand, camera, characters, cleaning, climates, controlled, creeping, crossings, crushed, decaying, demands, depicts, dominant, dressy, drunk, efficient, episode, everything, excellent, experience, family, firing, followed, forgiveness, freedom, fulfillment, functional, grazing, henceforth, ignored, illnesses, imitate, increasing, inevitable, introduced, January, materials, millionaires, mutineer, needed, obvious, package, paramagnetic, patiently, pleasant, possessed, pressure, radiation, recriminations, redecorating, rejected, secularist, shampooed, solid, spilled, spreader, story, strained, streamlined, street, stripped, stupid, superb, surface, swimming, sympathetically, unenthusiastic, unlined, urethane, usual, walking, weekday.

The keywords used in the HTIMIT experiments were: ambitious, appetite, avoided, bricks, building, causes, chroniclers, clinches, coeducational, colossal, concern, controlled, convincing, coyote, derived, desires, determination, disregarding, dwarf, effective, enrich, example, examples, excluded, executive, experiment, feverishly, firing, glossy, handle, happily, healthier, leaflet, lousiness, manure, misery, Nathan, northeast, notoriety, nutrients, obviously, overcame, penetrated, persuasively, petting, portion, precaution, prepare, prepared, privately, properties, propriety, reduced, referred, sandwich, sculptor, showering, sitting, sixty, sketched, skills, spirits, storm, strength, strip, surely, synagogue, technical, tomblike, traffic, tuna-fish, tycoons, university, vaguely, vanquished, virtues, waking, wedded, working, wounds.

The keywords used in WSJ experiment were: ability, administrative, analysis, answer, answer, business, business, children, children, clothes, company, confirmation, design, different, economy, economy, environment, environment, environment, equipment, evening, evening, experience, family, family, history, hospitalization, hospitalization, important, information, ingredients, interior, language, language, lesson, literature, literature, marketing, medicine, medicine, murder, murder, natural, necessary, newspaper, organizations, people, physical, physical, popular, popular, predispositions, preparation, private, procedure, process, progress, progress, psychological, public, public, questions, questions, reasons, regular, regular, research, research, responsibility, responsibility, scientists, sexual, simple, single, standards, strong, students, treatment, vegetable, violence.

The keywords used in OGI Stories experiment were: army, articles, available, baseball, boxing, bus, California, climbed, closed, competitive, contained, contributions, cost, course, crazy, creating, cutting, developed, directions, double, entertaining, experiencing, fee, fifty, forward, from, futures, Georgia, heading, innovation, institutional, interior, kick, kindness, land, listen, luck, Maine, main, much, never, nightly, nineteenth, operators, public, rancho,

recession, recommendations, Robert, room, such, teach, technology, Texas, though, turning, understood, ways, western, yesterday.

References

[1] L. Bahl, P. Brown, P. de Souza, and R. Mercer. Maximum mutual information estimation of hidden markov model parameters for speech recognition. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 49–52, 1986.

[2] Y. Benayed, D. Fohr, J.-P. Haton, and G. Chollet. Confidence measure for keyword spotting using support vector machines. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 588–591, 2004.

[3] S. Bengio, J. Maréthoz, and M. Keller. The expected performance curve. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

[4] H. Bourlard, B. D'Hoore, and J.-M. Boite. Optimizing recognition and rejection performance in wordspotting systems. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 373–376, 1994.

[5] P.S. Cardillo, M. Clements, and M.S. Miller. Phonetic searching vs. LVCSR: How to find what you really want in audio archives. *International Journal of Speech Technology*, 5:9–22, 2002.

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

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

[8] C. Cortes and M. Mohri. Confidence intervals for the area under the ROC curve. In *Advances in Neural Information Processing Systems 17*, 2004.

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

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

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

[12] O. Dekel, J. Keshet, and Y. Singer. Online algorithm for hierarchical phoneme classification. In *Workshop on Multimodal Interaction and Related Machine Learning Algorithms; Lecture Notes in Computer Science*, pages 146–159. Springer-Verlag, 2004.

[13] Q. Fu and B.-H. Juang. Automatic speech recognition based on weighted minimum classification error (W-MCE) training method. In *IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 278–283, 2007.

[14] B.-H. Juang, W. Chou, and C.-H. Lee. Minimum classification error rate methods for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 5(3):257–265, 1997.

[15] J. Junkawitsch, G. Ruske, and H. Höge. Efficient methods for detecting keywords in continuous speech. In *Proc. of European Conference on Speech Communication and Technology*, pages 259–262, 1997.

[16] J. Keshet, D. Chazan, and B.-Z. Bobrovsky. Plosive spotting with margin classifiers. In *Proceedings of the Seventh European Conference on Speech Communication and Technology*, pages 1637–1640, 2001.

[17] J. Keshet, S. Shalev-Shwartz, S. Bengio, Y. Singer, and D. Chazan. Discriminative kernel-based phoneme sequence recognition. In *Interspeech*, 2006.

[18] J. Keshet, S. Shalev-Shwartz, Y. Singer, and D. Chazan. A large margin algorithm for speech-to-phoneme and music-to-score alignment. *IEEE Trans. on Audio, Speech and Language Processing*, 15(8):2373–2382, 2007.

[19] H. Ketabdar, J. Vepa, S. Bengio, and H. Bourlard. Posterior based keyword spotting with a priori thresholds. In *Prof. of Interspeech*, 2006.

[20] A.S. Manos and V.W. Zue. A segment-based wordspotter using phonetic filler models. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 899–902, 1997.

[21] D.B. Paul and J.M. Baker. The design for the Wall Street Journal-based CSR corpus. In *Proc. of the International Conference on Spoken Language Processing*, 1992.

[22] J. C. Platt. Fast training of Support Vector Machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

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

[24] M.G. Rahim, C.H. Lee, and B.H. Juang. Discriminative utterance verification for connected digits recognition. *IEEE Transactions on Speech and Audio Processing*, pages 266–277, 1997.

[25] D.A. Reynolds. HTIMIT and LLHDB: speech corpora for the study of handset transducer effects. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 1535–1538, 1997.

[26] J. R. Rohlicek, P. Jeanrenaud, K. Ng H. Gish, B. Musicus, and M. Siu. Phonetic training and language modeling for word spotting. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 459–462, 1993.

[27] J. R. Rohlicek, William Russell, S. Roukod, and H. Gish. Continuous hidden markov model for speaker independent word spotting. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 627–430, 1989.

[28] R.C. Rose and D.B. Paul. A hidden markow model based keyword recognition system. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 129–132, 1990.

[29] J. Salomon, S. King, and M. Osborne. Framewise phone classification using support vector machines. In *Proceedings of the Seventh International Conference on Spoken Language Processing*, pages 2645–2648, 2002.

[30] S. Shalev-Shwartz, J. Keshet, and Y. Singer. Learning to align polyphonic music. In *Proceedings of the 5th International Conference on Music Information Retrieval*, 2004.

[31] M.-C. Silaghi and H. Bourlard. Iterative posterior-based keyword spotting without filler models. In *Proc. of the IEEE Automatic Speech Recognition and Understanding Workshop*, pages 213–216, Keystone, USA, 1999.

[32] I. Szoke, P. Schwarz, P. Matejka, L. Burget, M. Fapso, M. Karafiat, and J. Cernocky. Comparison of keyword spotting approaches for informal continuous speech. In *Proc. of Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, 2005.

[33] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Advances in Neural Information Processing Systems 17*, 2003.

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

[35] M. Weintraub. LVCSR log-likelihood ratio scoring for keyword spotting. In *Proc. of International Conference on Audio, Speech and Signal Processing*, pages 129–132, 1995.