Discriminative Keyword Spotting

Joseph Keshet [a,1,*], David Grangier [a], Samy Bengio [a,2]

[a] *IDIAP Research Institute, Rue du Simplon 4, CH-1920 Martigny, Switzerland*

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 a keyword spotter and discuss its formal properties. Experiments with the TIMIT corpus show that our method outperforms the conventional HMM-based approach. Further experiments using the TIMIT trained model, but tested on the WSJ dataset, show that without further training our method outperforms the conventional HMM-based approach.

Key words: Keyword spotting, Speech recognition, Large margin and kernel methods, Support vector machines

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

* Corresponding author.
Email addresses: jkeshet@cs.huji.ac.il (Joseph Keshet), grangier@idiap.ch (David Grangier), bengio@google.com (Samy Bengio).
[1] Present address: School of Computer Science and Engineering, The Hebrew University, Jerusalem 91904, Israel. Telephone: +972 54 6481306. Fax: +972 3 6489718.
[2] Present address: Google Inc., 1600 Amphitheatre Parkway, Mountain View, CA, 94043

based on hidden Markov models (HMMs). See for example (Benayed et al., 2004; Ketabdar et al., 2006; Silaghi and Bourlard, 1999; Szoke et al., 2005) 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 supervised learning and overcomes 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 Szoke et al., 2005), and does not estimate a garbage or background model (as in Silaghi and Bourlard, 1999). 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 Cristianini and Shawe-Taylor, 2000; Vapnik, 1998). One of the main goals of this work is to extend the notion of discriminative learning to the task of keyword spotting.

Our proposed method is based on recent advances in kernel machines and large margin classifiers for sequences (Shalev-Shwartz et al., 2004; Taskar et al., 2003), which in turn build on the pioneering work of Vapnik and colleagues (Cristianini and Shawe-Taylor, 2000; Vapnik, 1998). 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 (Keshet et al., 2001; Salomon et al., 2002). However, the model proposed in this paper is different from a classical SVM since we are not addressing a simple decision task such as binary classification or regression.

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 on Fig. 5. 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, i.e. *whole-word-modeling*, *phonetic-based* and *large-vocabulary-based* approaches.

Whole-word modeling is one of the earliest approaches using HMM for keyword spotting (Rohlicek et al., 1989; M.G. Rahim, 1997). 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 require 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 (Rohlicek et al., 1993; Bourlard et al., 1994; Manos and Zue, 1997). 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 (Rabiner and Juang, 1993). 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. Bourlard et al. (1994); Manos and Zue (1997). A third 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 Junkawitsch et al. (1997).

A further extension of HMM spotter approaches consists in 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, excepted the targeted keyword. This use of additional linguistic constraints is shown to improve the spotting performance (Rose and Paul, 1990; Weintraub, 1995; P. S. Cardillo and Miller, 2002; Szoke et al., 2005). Such an approach however raises some conceptual and practical concerns. From a conceptual point of view, one can wonder whether an automatic system should require such a linguistic knowledge while a human address the keyword spotting task without knowing a large vocabulary in the targeted language. Besides this aspect, one can also wonder whether the design of a keyword spotting should require the expensive collection a large amount of labeled data typically needed to train LVCSR systems, as well as

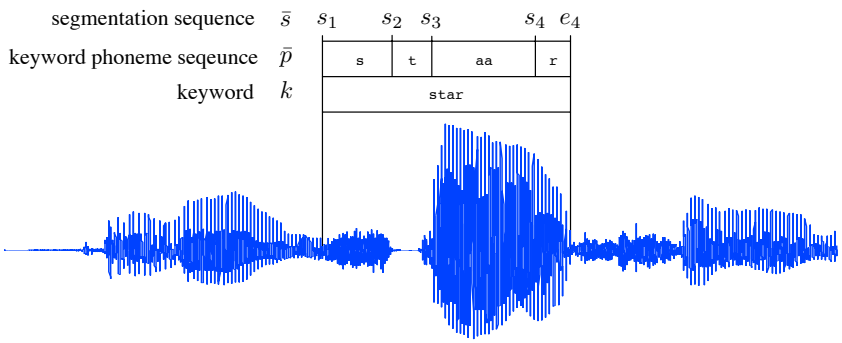

Fig. 1. Example of our notation. The waveform of the spoken utterance "a lone star shone..." taken from the TIMIT corpus. The keyword k is the word *star*. The phonetic transcription $\bar{p}$ along with the alignment sequence $\bar{s}$ are schematically depicted in the figure.

the computational requirement to perform large vocabulary decoding (Manos and Zue, 1997).

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. Our method is based on non-linear phoneme recognition and segmentation functions. The specific feature functions we use for are presented in Sec. 5. In Sec. 6 we present experimental results with the TIMIT corpus and with the Wall Street journal (WSJ) corpus. We conclude the paper in Sec. 7.

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 decide whether the keyword is uttered or not, namely, whether the corresponding sequence of phonemes is articulated in the given utterance.

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 $\mathcal{P}$. 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 to actually determine whether $\bar{p}^k$ is uttered in $\bar{\mathbf{x}}$. Let us further define the alignment of a phoneme sequence to 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 alignment sequence $\bar{s}^k$ 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.

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 Benayed et al., 2004; Ketabdar et al., 2006; Silaghi and Bourlard, 1999). 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 different cost functions weighing false positive and false negative errors. Assuming a flat prior over all these cost functions, a criterion to identify a good keyword spotting system that would be good on average for all these settings could be to select the one 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 supervised algorithm for learning a spotting function f from a training set of examples. Our construction is based on a set of predefined feature functions $\{\phi\}_{j=1}^{n}$. 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 alignment sequence $\bar{s}^k \in \mathbb{N}^*$ into an abstract vector-space, and returns a scalar in $\mathbb{R}$ which, intuitively, represents the confidence in the suggested alignment sequence given the keyword phoneme sequence $\bar{p}^k$. For example, one element of the feature function can sum the number of times phoneme p comes after phoneme p', while other elements of the 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. 5.

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 function elements over all possible alignment sequences. The maximization defined by Eq. (1) is over an exponentially large number of alignment sequences. Nevertheless, as in HMMs, if the feature functions $\boldsymbol{\phi}$ are decomposable, the maximization in Eq. (1) can be efficiently calculated using a dynamic programming procedure.

Recall that we would like to obtain a system that maximizes the AUC on unseen data. In order to do so, we will maximize the AUC over a large set of training examples. In Appendix A we show that our algorithm which maximizes the AUC over the training set is likely to maximizes the AUC over an unseen data as well. Let us consider two sets of 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* (Cortes and Mohri, 2004) 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 $\mathbb{1}_{\{\cdot\}}$ 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 an alignment sequence $\bar{s}^k$ that corresponds to the location of the keyword in $\bar{\mathbf{x}}^+$. Overall we have m examples, that is, $S = \{(\bar{p}^{k_1}, \bar{\mathbf{x}}_1^+, \bar{\mathbf{x}}_1^-, \bar{s}_1^{k_1}), \ldots, (\bar{p}^{k_m}, \bar{\mathbf{x}}_m^+, \bar{\mathbf{x}}_m^-, \bar{s}_m^{k_m})\}$. We assume that we have access to

the correct alignment $\bar{s}^k$ of the phonemes sequence $\bar{p}^k$ for each training utterance $\bar{\mathbf{x}}^+ \in \mathcal{X}_k^+$. This assumption is actually not restrictive since such an alignment can be inferred relying on an alignment algorithm (Keshet et al., 2005).

Similar to the SVM algorithm for binary classification (Cortes and Vapnik, 1995; Vapnik, 1998), 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 alignment sequence $\bar{s}$. Second, we find a vector $\mathbf{w} \in \mathbb{R}^n$, such that the projection of vectors onto $\mathbf{w}$ ranks the vectors constructed in the first step above 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

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

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 even consider the best alignment 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 alignment 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 (Vapnik, 1998).

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 (Cortes and Vapnik, 1995; Vapnik, 1998) and define the following hinge-loss function,

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

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 Cristianini and Shawe-Taylor, 2000). 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 (Platt, 1998), 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 (Crammer et al., 2006).

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 alignment 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 normalized 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,

$$\Delta\boldsymbol{\phi}_i = \frac{1}{|\mathcal{X}_{k_i}^+||\mathcal{X}_{k_i}^-|} \left(\boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}^{k_i}) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s}') \right) \ . \tag{8}$$

The normalization factor is a result of our goal to minimize the average AUC (see Appendix A). We set the next weight vector $\mathbf{w}_i$ to be the minimizer of the following optimization problem,

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

where C serves as a complexity-accuracy trade-off parameter (see Crammer et al. (2006)) and ξ is a non-negative slack variable, which indicates the loss of the ith example. Intuitively, we would like to 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

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}.

Fig. 2. An iterative algorithm.

the projection of the sequence that does not contains it onto $\mathbf{w}$ by at least 1. It can be shown (see Crammer et al., 2006) 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 ongoing work on online learning algorithms appearing in (Crammer et al., 2006). Based on this work, it is shown in Appendix A that, under some mild technical conditions, the cumulative performance 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, it can further be shown (see Appendix A) that if the cumulative performance of the iterative procedure is high, there exists at least one weight vector among the vectors $\{\mathbf{w}_1, \ldots, \mathbf{w}_m\}$ which attains high averaged performance on unseen examples as well, that is, there exists a vector which attains high averaged AUC over a set of unseen examples. To find this 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 Bengio et al., 2005) 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 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 alignment sequence 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 speech phonetic segmentation (Keshet et al., 2005).

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 an alignment 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

$$\phi_j(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=2}^{|\bar{p}|-1} d(\mathbf{x}_{-j+s_i}, \mathbf{x}_{j+s_i}), \quad j \in \{1, 2, 3, 4\} \ . \tag{12}$$

If $\bar{s}$ is the correct timing sequence 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 Dekel et al. (2004). 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,

$$\phi_5(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=1}^{|\bar{p}|} \sum_{t=s_i}^{s_{i+1}-1} g_{p_i}(\mathbf{x}_t) \ . \tag{13}$$

Our next feature function scores alignment 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,

$$\phi_6(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=1}^{|\bar{p}|} \log \ \mathcal{N}(s_{i+1} - s_i; \hat{\mu}_{p_i} \ , \hat{\sigma}_{p_i}) \quad , \tag{14}$$

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. 6).

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,

$$\phi_7(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=2}^{|\bar{p}|} (r_i - r_{i-1})^2 \quad . \tag{15}$$

Each of the feature functions is normalized by the number of frames in the speech utterance, and each of the feature functions is weighted be a fixed constant, $\{\beta_j\}_{j=1}^{7}$. The constants are determined so as to maximize performance over a validation set.

6 Experimental Results

To validate the effectiveness of the proposed approach we performed experiments with the TIMIT corpus. We divided the training portion of TIMIT (ex-

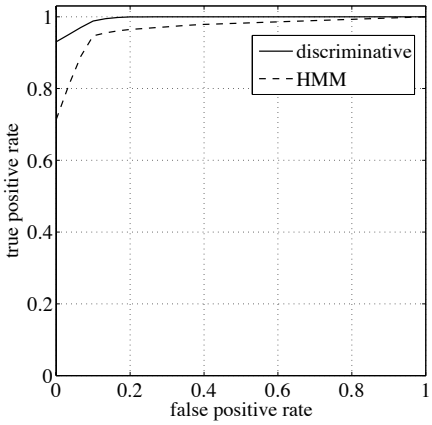

Fig. 3. ROC curves of the discriminative algorithm and the HMM approach, trained on the TIMIT training set and tested on 80 keywords from TIMIT test set. The AUC of the ROC curves is **0.99** and **0.96** for the discriminative algorithm and the HMM algorithm, respectively.

cluding the SA1 and SA2 utterances) into three disjoint parts containing 500, 80 and 3116 utterances. The first part of the training set was used for learning the functions g_p (Eq. (13)), which define the feature function ϕ_5. Those functions were learned by the algorithm described in Dekel et al. (2004) using the MFCC+Δ+$\Delta\Delta$ acoustic features and a Gaussian kernel with parameter $\sigma = 6.24$.

The second set of 80 utterances formed the validation set needed for our keyword spotting algorithm. The set was built out of a set of 40 keywords randomly chosen from the TIMIT lexicon. The 80 utterances were chosen by pairs: one utterance in which the keyword was uttered and another utterance in which the keyword was not uttered. Finally, we ran our iterative algorithm on the rest of the utterances in the training set. The value of the parameter C was set to be 1.

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 (Collobert et al., 2002). All hyper-parameters including number of states, number of Gaussians per state, variance flooring factor, were tuned using the validation set.

Keyword detection was performed with a new HMM composed of two sub

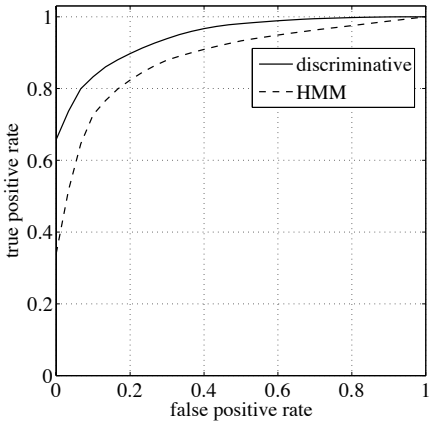

Fig. 4. ROC curves of the discriminative algorithm and the HMM approach, trained on the TIMIT training set and tested on 80 keywords from WSJ test set. The AUC of the ROC curves is **0.94** and **0.88** for the discriminative algorithm and the HMM algorithm, respectively.

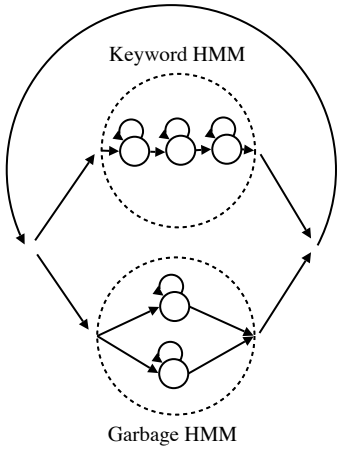

Fig. 5. HMM topology for keyword spotting.

HMM models, the keyword model and the garbage model, as depicted in Fig. 5. 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.

Table 1
The AUC of the discriminative algorithm compared to the HMM in the experiments.

	Discriminative Algo.	HMM
Corpus	AUC	AUC
TIMIT	**0.99**	0.96
WSJ	**0.94**	0.88

The test set was composed of 80 randomly chosen keywords, distinct from the keywords of the training and validation sets. 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 was evaluated against the test data. The results are reported as averaged ROC curves in Fig. 3. The AUC of the ROC curves is 0.99 and 0.96 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.

The next experiment examines the robustness of the proposed algorithm. We compared the performance of the proposed discriminative algorithm and of the HMM on the WSJ corpus (Paul and Baker, 1992). Both systems were trained on the TIMIT corpus as describe above and tested on the same 80 keywords. For each keyword we randomly picked at most 20 utterances from the `si_tr_s` portion of the WSJ corpus. The ROC curves are given in Fig. 4. The AUC of the ROC curves is 0.94 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.

A summary of the results of both experiments is given in Table 1. Close look on both experiments, we see that the discriminative algorithm outperforms the HMM in terms of AUC. This indeed validates our theoretical analysis that our algorithm maximizes the AUC. Moreover, the discriminative algorithm outperforms the HMM in all point of the ROC curve, meaning that it has better true positive rate for every given false negative rate.

7 Conclusions

Keyword spotting is a speech related task with more and more practical interest from an application point of view. Nevertheless, current state-of-the-art approaches are still 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 state-of-the-art approaches which mostly rely on generative HMM models, the proposed model has shown to yield a statistically significant improvement over the TIMIT corpus. Furthermore, the very same model trained on the TIMIT corpus but now tested on the WSJ corpus also yielded a statistically significantly better performance than the HMM based approach. Various extensions of this approach can be foreseen. For instance...

A Theoretical Analysis

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

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

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,*

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

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

Proof The proof of the theorem relies on Lemma 1 and Theorem 4 in (Crammer et al., 2006). Lemma 1 in (Crammer et al., 2006) implies that,

$$\sum_{i=1}^{m} \alpha_i \left(2\ell_i - \alpha_i \|\Delta\boldsymbol{\phi}_i\|^2 - 2\ell_i^\star\right) \leq \|\mathbf{w}^\star\|^2. \tag{A.3}$$

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

$$\alpha_i \ell_i \geq \min\left\{\frac{\ell_i}{\Delta\boldsymbol{\phi}_i}, C\right\} \geq \min\{1, C\} = 1. \tag{A.4}$$

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

$$\sum_{i=1}^{m} \alpha_i \ell_i \geq \sum_{i=1}^{m} \mathbb{1}_{\{\mathbf{w}_i \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}^{k_i}) \leq \mathbf{w}_i \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}^-, \bar{p}^{k_i}, \bar{s}_i')\}}. \tag{A.5}$$

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. (A.5) into Eq. (A.3) we get

$$\sum_{i=1}^{m} \mathbb{1}_{\{\mathbf{w}_i \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}^{k_i}) \leq \mathbf{w}_i \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}^-, \bar{p}^{k_i}, \bar{s}_i')\}} \leq \|\mathbf{w}^\star\|^2 + 2C \sum_{i=1}^{m} \ell_i^\star. \tag{A.6}$$

The theorem follows by replacing the sum over prediction mistakes to a sum over prediction hits and pluging-in the definition of the cumulative AUC given in Eq. (A.1). □

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*

$$1 - \hat{A} = \mathbb{E}_Q\left[\mathbb{1}_{\{f(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) \leq f(\bar{\mathbf{x}}^-, \bar{p}^{k_i})\}}\right] = \Pr_Q\left[f(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}) \leq f(\bar{\mathbf{x}}^-, \bar{p}^{k_i})\right] \leq$$
$$\frac{1}{m}\sum_{i=1}^{m} \ell(\mathbf{w}^\star; (\bar{p}^{k_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{s}^{k_i})) + \frac{\|\mathbf{w}^\star\|^2}{m} + \frac{\sqrt{2\ln(2/\delta)}}{\sqrt{m}} + \frac{\sqrt{2\ln(2m/\delta)}}{\sqrt{m_{\text{val}}}} \quad , \tag{A.7}$$

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

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

Proposition 1 in (Cesa-Bianchi et al., 2004) implies that with probability of at least $1 - \delta_1$ the following bound holds,

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

Combining this fact with Thm. 1 we obtain that,

$$\frac{1}{m}\sum_{i=1}^{m} \mathrm{risk}(f_i) \;\leq\; \frac{2C}{m}\sum_{i=1}^{m} \ell_i^\star + \frac{\|\mathbf{w}^\star\|^2}{m} + \frac{\sqrt{2\ln\left(1/\delta_1\right)}}{\sqrt{m}} \quad . \tag{A.8}$$

The left-hand side of the above inequality upper bounds $\mathrm{risk}(f^\star)$, where $f^\star = \arg\min_{f_i} \mathrm{risk}(f_i)$. 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. (A.8). 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 $1 - \delta_2$,

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

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

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

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

Acknowledgements

This research was done while Joseph Keshet was visiting IDIAP Research Institute. This research was supported by the European PASCAL Network of Excellence and the DIRAC project.

References

Benayed, Y., Fohr, D., Haton, J.-P., Chollet, G., 2004. Confidence measure for keyword spotting using support vector machines. In: Proc. of International Conference on Audio, Speech and Signal Processing.

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

Bourlard, H., DHoore, B., Boite, J.-M., 1994. Optimizing recognition and rejection performance in wordspotting systems. In: Proc. of International Conference on Audio, Speech and Signal Processing. pp. 373–376.

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

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

Cortes, C., Mohri, M., 2004. Confidence intervals for the area under the roc curve. In: Advances in Neural Information Processing Systems 17.

Cortes, C., Vapnik, V., September 1995. Support-vector networks. Machine Learning 20 (3), 273–297.

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

Cristianini, N., Shawe-Taylor, J., 2000. An Introduction to Support Vector Machines. Cambridge University Press.

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

Junkawitsch, J., Ruske, G., Hge, H., 1997. Efficient methods for detecting keywords in continuous speech. In: Proc. of European Conference on Speech Communication and Technology. pp. 259–262.

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

Keshet, J., Shalev-Shwartz, S., Singer, Y., Chazan, D., 2005. Phoneme alignment based on discriminative learning. In: Interspeech.

Ketabdar, H., Vepa, J., Bengio, S., Bourlard, H., 2006. Posterior based keyword spotting with a priori thresholds. In: Proceeding of Interspeech.

Manos, A., Zue, V., 1997. A segment-based wordspotter using phonetic filler models. In: Proc. of International Conference on Audio, Speech and Signal Processing. pp. 899–902.

M.G. Rahim, C.H. Lee, B. J., 1997. Discriminative utterance verification for connected digits recognition. IEEE Transactions on Speech and Audio Processing, 266–277.

P. S. Cardillo, M. C., Miller, M. S., 2002. Phonetic searching vs. lvcsr: How to find what you really want in audio archives. International Journal of Speech Technology 5, 9–22.

Paul, D., Baker, J., 1992. The design for the Wall Street Journal-based CSR corpus.

In: Proc. of ICSLP.

Platt, J. C., 1998. Fast training of Support Vector Machines using sequential minimal optimization. In: Schölkopf, B., Burges, C., Smola, A. (Eds.), Advances in Kernel Methods - Support Vector Learning. MIT Press.

Rabiner, L., Juang, B., 1993. Fundamentals of speech recognition. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

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

Rohlicek, J. R., Russell, W., Roukod, S., Gish, H., 1989. Continuous hidden markov model for speaker independent word spotting. In: Proc. of International Conference on Audio, Speech and Signal Processing. pp. 627–430.

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

Salomon, J., King, S., Osborne, M., 2002. Framewise phone classification using support vector machines. In: Proceedings of the Seventh International Conference on Spoken Language Processing. pp. 2645–2648.

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

Silaghi, M.-C., Bourlard, H., 1999. Iterative posterior-based keyword spotting without filler models. In: Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop. Keystone, USA, pp. 213–216.

Szoke, I., Schwarz, P., Matejka, P., Burget, L., Fapso, M., Karafiat, M., Cernocky, J., 2005. Comparison of keyword spotting approaches for informal continuous speech. In: Proc. of MLMI.

Taskar, B., Guestrin, C., Koller, D., 2003. Max-margin markov networks. In: Advances in Neural Information Processing Systems 17.

Vapnik, V. N., 1998. Statistical Learning Theory. Wiley.

Weintraub, M., 1995. Lvcsr log-likelihood ratio scoring for keyword spotting. In: Proc. of International Conference on Audio, Speech and Signal Processing. pp. 129–132.