Discriminative Keyword Spotting

David Grangier*, Joseph Keshet†, Samy Bengio‡

Abstract

This chapter introduces a discriminative method for detecting and spotting keywords in spoken utterances. Given a word represented as a sequence of phonemes and a spoken utterance, the keyword spotter predicts the best time span of the phoneme sequence in the spoken utterance along with a confidence. If the prediction confidence is above certain level the keyword is declared to be spoken in the utterance within the predicted time span, otherwise the keyword is declared as not spoken. The problem of keyword spotting training is formulated as a discriminative task where the model parameters are chosen so the utterance in which the keyword is spoken would have higher confidence than any other spoken utterance in which the keyword is not spoken. It is shown theoretically and empirically that the proposed training method resulted with a high area under the receiver operating characteristic (ROC) curve, the most common measure to evaluate keyword spotters. We present an iterative algorithm to train the keyword spotter efficiently. The proposed approach contrasts with standard spotting strategies based on HMMs, for which the training procedure does not maximize a loss directly related to the spotting performance. Several experiments performed on TIMIT and WSJ corpora show the advantage of our approach over HMM-based alternatives.

1 Introduction

Keyword spotting aims at detecting any given keyword in spoken utterances. This task is important in numerous applications, such as voice mail retrieval, voice command detection and spoken term detection and retrieval. Previous work has focused mainly on several variants of Hidden Markov Models (HMMs) to address this intrinsically sequential problem. While the HMM-based approaches constitute the state-of-the-art, they suffer from several known limitations. Most of these limitations are not specific to the keyword spotting problem, and are common to other tasks such as speech recognition, as pointed out in [19]. For instance, the predominance of the emission probabilities in the likelihood, which tends to neglect duration and transition models, or the

*NEC Laboratories America, Princeton, NJ, USA
†IDIAP Research Institute, Martigny, Switzerland
‡Google Inc., Mountain View, CA, USA

Expectation-Maximization (EM) training procedure, which is prone to convergence to local optima. Other drawbacks are specific to the application of HMMs to the keyword spotting task. In particular, the scarce occurrence of some keywords in the training corpora often requires ad-hoc modifications of the HMM topology, the transition probabilities or the decoding algorithm. The most acute limitation of HMM-based approaches lies in their training objective. Typically, HMM training aims at maximizing the likelihood of transcribed utterances, and does not provide any guarantees in terms of keyword spotting performance.

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 [3, 20, 29]. Each point on the ROC curve represents the system performance for a specific trade-off between achieving a high true positive rate and a low false positive rate. Since the preferred trade-off is not always defined in advance, systems are commonly evaluated according to the averaged performance over all operating points. This corresponds to preferring the systems that attain the highest Area Under the ROC Curve (AUC).

In this study, we devise a discriminative large margin approach for learning to spot any given keyword in any given speech utterance. The keyword spotting function gets as input a phoneme sequence representing the keyword and a spoken utterance and outputs a prediction of the time span of the keyword in the spoken utterance and a confidence. If the confidence is above some predefined threshold, the keyword is declared to be spoken in the predicted time span, otherwise the keyword is declared as not spoken. The goal of the training algorithm is to maximize the AUC on the training data and on unseen test data. We call an utterance in the training set in which the keyword is spoken a *positive utterance*, and respectively, an utterance in which the keyword is not spoken a *negative utterance*. Using the Wilcoxon-Mann-Whitney statistics [8], we formulate the training as a problem of estimating the model parameters such that the confidence of the correct time span in a positive utterance would be higher than the confidence of any time span in any negative utterance. Formally this problem is stated as a convex optimization problem with constraints. The solution to this optimization problem is a function which shown analytically to attain high AUC on the training set and is likely to have good generalization properties on unseen test data as well. Moreover, comparing to HMMs, our approach is based on a convex optimization procedure, which converges to the global optima, and it is based on non-probabilistic framework, which offers greater flexibility in selecting the relative importance of duration modeling with respect to acoustic modeling.

The remainder of this chapter is organized as follows: Section 2 describes previous work on keyword spotting, Section 3 introduces our discriminative large margin approach, Section 4 presents different experiments comparing the proposed model to an HMM-based solution. Finally, Section 5 draws some conclusions and delineates possible directions for future research.

2 Previous Work

The research on keyword spotting has paralleled the development of the Automatic Speech Recognition (ASR) domain in the last thirty years. Like ASR, keyword spotting has first been addressed with models based on Dynamic Time Warping (DTW) [6, 14]. Then, approaches based on discrete HMMs have been introduced [18]. Finally, discrete HMMs have been replaced by continuous HMMs [25].

The core objective of a keyword spotting system is to discriminate between utterances in which a given keyword is uttered to utterances in which the keyword is not uttered. For this purpose, the first approaches based on DTW proposed to compute the alignment distance between a template utterance representing the target keyword and all possible segments of the test signal [6]. In this context, the keyword is considered as detected in a segment of the test utterance whenever the alignment distance is below some predefined threshold. Such approaches are however greatly affected by speaker mismatch and varying recording conditions between the template sequence and the test signal. To gain some robustness, it has then been proposed to compute alignment distances not only with respect to the target keyword template, but also with respect to other keyword templates [14]. Precisely, given a test utterance, the system identifies the concatenation of templates with the lowest distance to the signal and the keyword is considered as detected if this concatenation contains the target keyword template. Therefore, the keyword alignment distance is not considered as an absolute number, but relatively to the distances to other templates, which increase robustness with respect to changes in the recording conditions.

Along with the development of the speech research, increasingly large amount of labeled speech data were collected, and DTW-based techniques started showing their shortcomings to leverage from large training sets. In particular, large corpora contains thousands of words and dozens of instances for each word. Considering each instance as a template makes the search for the best template concatenation a prohibitively expensive task. Consequently, discrete HMMs were introduced for ASR [1], and then for keyword spotting [18, 35]. A discrete HMM assumes that the quantized acoustic feature vectors representing the input utterance are independent conditioned on the hidden state variables. This type of model introduces several advantages compared to DTW-based approaches, including an improved robustness to speaker and channel changes, when several training utterances of the targeted keyword are available. However, the most important evolution introduced with the HMM certainly lies in the development of phone or triphone-based modeling [21, 18, 27], in which a word model is composed of several sub-unit models shared across words. This means that the model of a given word not only benefits from the training utterances containing this word, but also from all the utterances containing its sub-units. Additionally, great scalability is achieved with such an approach since the number of acoustic parameters is proportional to the number of sub-units and does not grow linearly with the corpus size, as opposed to template-based approaches. A further advantage of phone-based modeling is the ability to spot words unavail-

able at training time, as this paradigm allows one to build a new word model by composing already trained sub-unit models. This aspect is very important, since in most applications the set of test keywords is not known in advance.

Soon after the application of discrete HMMs to speech problems, continuous density HMMs have been introduced in the ASR community [25]. Continuous HMMs eliminate the need of acoustic vector quantization, as the distributions associated with the HMM states are continuous densities, often modeled by Gaussian Mixtures Models (GMMs). The learning of both GMM parameters and the state transition probabilities is performed in a single integrated framework, maximizing the likelihood of the training data given its transcription through the Expectation-Maximization (EM) algorithm [4]. This approach has been shown to be more effective and allows greater flexibility for speaker or channel adaptation [25]. It is now the most widely used approach for both ASR and keyword spotting.

In the context of keyword spotting, different strategies based on continuous HMMs have been proposed. In most cases, a sub-unit based HMM is trained over a large corpus of transcribed data and a new model is then built from the sub-unit models. Such a model is composed of two parts, a keyword HMM and a *filler* or *garbage* HMM, which respectively model the keyword and non-keyword parts of the signal. This topology is depicted in Figure 1. Given such a model, keyword detection is performed by searching for the sequence of states that yields the highest likelihood for the provided test sequence through Viterbi decoding. Keyword detection is determined by checking whether the Viterbi best-path passes through the keyword model or not. In such a model, the selection of the transition probabilities in the keyword sets the trade-off between low false alarm rate (detecting a keyword when it is not presented), and low false rejection rate (not detecting a keyword when it is indeed presented). Another important aspect of this approach lies in the modeling of non-keyword parts of the signal, and several choices are possible for the garbage HMM. The simplest choice models garbage with an HMM that fully connects all sub-units models [27], while the most complex choice models garbage with a full-large vocabulary HMM, where the lexicon excludes the keyword [31]. The latter approach obviously yields a better garbage model, using additional linguistic knowledge. This advantage however induces a higher decoding cost and requires larger amount of training data, in particular for language model training. Besides practical concerns, one can conceptually wonder whether an automatic spotting approach should re-

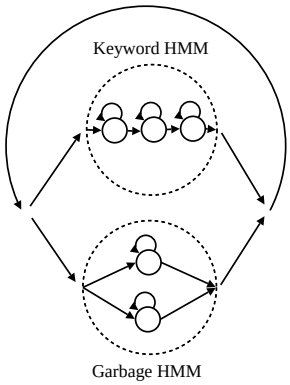

Figure 1: HMM topology for keyword spotting with a Viterbi best path strategy. This approach verifies whether the Viterbi best path passes through the keyword submodel.

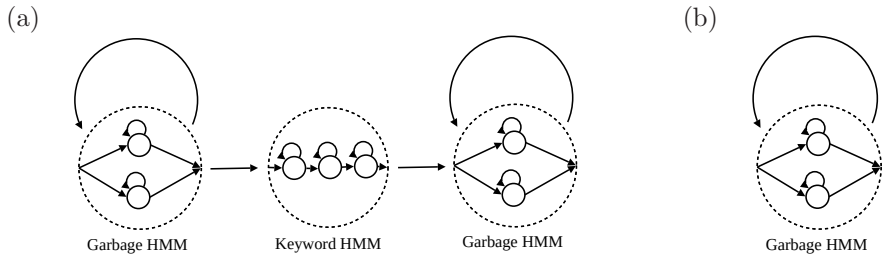

Figure 2: HMM topology for keyword spotting with a likelihood ratio strategy. This approach compares the likelihood of the sequence given the keyword is uttered (a), to the likelihood of the sequence given the keyword is not uttered (b).

quire such a large linguistic knowledge. Of course, several variations of garbage models exist between the two extreme examples pointed above (see for instance [5]).

Viterbi decoding relies on a sequence of local decisions to determine the best path, which can be fragile with respect to local model mismatch. In the context of HMM-based keyword spotting, a keyword can be missed, if only its first phoneme suffers such a mismatch, for instance. To gain some robustness, likelihood ratio approaches have been proposed [32, 27]. In this case, the confidence score outputted from the keyword spotter corresponds to the ratio between the likelihood estimated by an HMM including the occurrence of the target keyword, and the likelihood estimated by an HMM excluding it. These HMM topologies are depicted in Figure 2. Detection is then performed by comparing the outputted scores to a predefined threshold. Different variations on this likelihood ratio approach have then been devised, such as computing the ratio only on the part of the signal where the keyword is assumed to be detected [17]. Overall, all the above methods are variations over the same HMM paradigm, which consists in training a generative model through likelihood maximization, before introducing different modifications prior to decoding in order to address the keyword spotting task. In other words, these approaches do not propose to train the model so as to maximize the spotting performance, and the keyword spotting task is only introduced in the inference step after training.

Only few studies have proposed discriminative parameter training approaches to circumvent this weakness [30, 28, 33, 2]. [30] proposed to maximize the likelihood ratio between the keyword and garbage models for keyword utterances and to minimize it over a set of false alarms generated by a first keyword spotter. [28] proposed to apply Minimum Classification Error (MCE) to the keyword spotting problem. The training procedure updates the acoustic models to lower the score of non-keyword models in the part of the signal where the keyword is uttered. However, this procedure does not focus on false alarms, and does not aim at lowering the score of the keyword-models in parts of the signal where the

keyword is not uttered. Other discriminative approaches have been focused on combining different HMM-based keyword detectors. For instance, [33] trained a neural network to combine likelihood ratios from different models. [2] relied on support vector machines to combine different averages of phone-level likelihoods. Both of these approaches propose to minimize the error rate, which equally weights the two possible spotting errors, false positive (or false alarm) and false negative (missed keyword occurrence, often called keyword deletion). This measure is however barely used to evaluate keyword spotters, due to the *unbalanced* nature of the problem. Precisely, the targeted keywords generally occurs rarely and hence the number of potential false alarms highly exceeds the number of potential missed detections. In this case, the useless model which never predicts the keyword avoids all false alarms and yields a very low error rate, with which it is difficult to compete. For that reason the AUC is more informative and is commonly used to evaluate models. Attaining high AUC would hence be an appropriate learning objective for the discriminative training of a keyword spotter. To the best of our knowledge, only [7] proposed an approach targeting this goal. This work introduces a methodology to maximize the Figure-Of-Merit (FOM), which corresponds to the AUC over a specific range of false alarm rates. However, the proposed approach relies on various heuristics, such as gradient smoothing and sorting approximations, which does not ensure any theoretical guarantee on obtaining high FOM. Also, these heuristics involve the selection of several hyperparameters, that challenges a practical use.

In the following, we introduce a model that aims at achieving high AUC over a set of training examples, and constitutes a truly discriminative approach to the keyword spotting problem. The proposed model relies on large margin learning techniques for sequence prediction and provides theoretical guarantees regarding the generalization performance. Furthermore, its efficient learning procedure ensures scalability toward large problems and simple practical use.

3 Discriminative Keyword Spotting

This section formalizes the keyword spotting problem, and introduces the proposed approach. First, we describe the problem of keyword spotting formally. This allows us to introduce a loss derived from the definition of the AUC. Then, we present our model parameterization and the training procedure to minimize efficiently a regularized version of this loss. Finally, we give an analysis of the iterative algorithm, and show it achieves a high cumulative AUC in the training process and high expected AUC on unseen test data.

3.1 Problem Setting

In the keyword spotting task, we are provided with a speech signal composed of a sequence of acoustic feature vectors $\overline{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathcal{X} \subset \mathbb{R}^d$, for all $1 \leq t \leq T$, is a feature vector of length d extracted from the t-th frame. Naturally, the length of the acoustic signal varies from one signal to another and

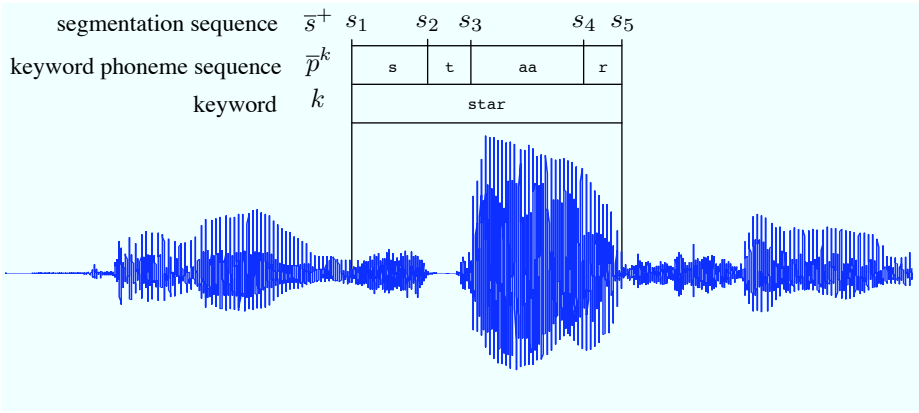

Figure 3: Example of our notation. The waveform of the spoken utterance "a lone star shone..." taken from the TIMIT corpus. The keyword k is the word *star*. The phonetic transcription $\overline{p}^k$ along with the time-span sequence $\overline{s}^+$ are depicted in the figure.

thus T is not fixed. 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 $\overline{p}^k = (p_1, \ldots, p_L)$, where $p_l \in \mathcal{P}$ for all $1 \leq l \leq L$ and $\mathcal{P}$ is the domain of the phoneme symbols. The number of phonemes in each keyword may vary from one keyword to another and hence L is not not fixed. We denote by $\mathcal{P}^*$ (and similarly $\mathcal{X}^*$) the set of all finite length sequences over $\mathcal{P}$. Let us further define the time span of the phoneme sequence $\overline{p}^k$ in the speech signal $\overline{\mathbf{x}}$. We denote by $s_l \in \{1, \ldots, T\}$ the start time (in frame units) of phoneme p_l in $\overline{\mathbf{x}}$, and by $e_l \in \{1, \ldots, T\}$ the end time of phoneme p_l in $\overline{\mathbf{x}}$. We assume that the start time of any phoneme p_{l+1} is equal to the end time of the previous phoneme p_l, that is, $e_l = s_{l+1}$ for all $1 \leq l \leq L-1$. We define the time span (or segmentation) sequence as $\overline{s}^k = (s_1, \ldots, s_L, e_L)$. An example of our notation is given in Figure 3. Our goal is to learn a *keyword spotter*, denoted $f : \mathcal{X}^* \times \mathcal{P}^* \rightarrow \mathbb{R}$, which takes as input the pair $(\overline{\mathbf{x}}, \overline{p}^k)$ and returns a real valued score expressing the confidence that the targeted keyword k is uttered in $\overline{\mathbf{x}}$. By comparing this score to a threshold $b \in \mathbb{R}$, we can determine whether $\overline{p}^k$ is uttered in $\overline{\mathbf{x}}$.

In discriminative supervised learning we are provided with a training set of examples and a test set (or evaluation set). Specifically, in the task of discriminative keyword spotting we are provided with a two sets of keywords. The first set $\mathcal{K}_{\text{train}}$ is used for training and the second set $\mathcal{K}_{\text{test}}$ is used for evaluation. Note that the lexicon of keywords is a union of both the training set and the test set, $\mathcal{K} = \mathcal{K}_{\text{train}} \cup \mathcal{K}_{\text{test}}$. Algorithmically, we do not restrict a keyword to be only in one set and a keyword that appears in the training set can appear also in the test set. Nevertheless, in our experiments we picked different keywords for training and test and hence $\mathcal{K}_{\text{train}} \cap \mathcal{K}_{\text{test}} = \emptyset$.

A keyword spotter f is often evaluated using the ROC curve. This curve plots the true positive rate (TPR) as a function of the false positive rate (FPR). The TPR measures the fraction of keyword occurrences correctly spotted, while the FPR measures the fraction of negative utterances yielding a false alarm. The points on the curve are obtained by sweeping the threshold b from the largest value outputted by the system to the smallest one. These values hence correspond to different trade-offs between the two types of errors a keyword spotter can make, i.e., missing a keyword utterance or rising a false alarm. In order to evaluate a keyword spotter over various trade-offs, it is common to report the AUC as a single value. The AUC hence corresponds to an averaged performance, assuming a flat prior over the different operational settings. Given a keyword k, a set of positive utterances X_k^+ in which k is uttered, and a set of negative utterances X_k^- in which k is not uttered, the AUC can be written as,

$$A_k = \frac{1}{|X_k^+||X_k^-|} \sum_{\overline{\mathbf{x}}^+ \in X_k^+} \sum_{\overline{\mathbf{x}}^- \in X_k^-} \mathbb{1}_{f(\overline{p}^k, \overline{\mathbf{x}}^+) > f(\overline{p}^k, \overline{\mathbf{x}}^-)} \ ,$$

where $|\cdot|$ refers to set cardinality and $\mathbb{1}_\pi$ refers to the indicator function and its value is 1 if the predicate π holds and 0 otherwise. The AUC of the keyword k, A_k, hence estimates the probability that the score assigned to a positive utterance is greater than the score assigned to a negative utterance. This quantity is also referred to as the *Wilcoxon-Mann-Whitney statistics* [34, 23, 8].

As one is often interested in the expected performance over any keyword, it is common to plot the ROC averaged over a set of evaluation keywords $\mathcal{K}_{\text{test}}$, and to compute the corresponding averaged AUC,

$$A_{\text{test}} = \frac{1}{|\mathcal{K}_{\text{test}}|} \sum_{k \in \mathcal{K}_{\text{test}}} A_k.$$

In this study, we introduce a large-margin approach to learn a keyword spotter f from a training set, which achieves a high averaged AUC.

3.2 Loss Function and Model Parameterization

In order to build our keyword spotter f, we are given training data consisting of a set of training keywords $\mathcal{K}_{\text{train}}$ and a set of training utterances. For each keyword $k \in \mathcal{K}_{\text{train}}$, we denote with X_k^+ the set of utterances in which the keyword is spoken and with X_k^- the set of all other utterances, in which the keyword is not spoken. Furthermore, for each positive utterance $\overline{\mathbf{x}}^+ \in X_k^+$, we are also given the timing sequence $\overline{s}^+$ of the keyword phoneme sequence $\overline{p}^k$ in $\overline{\mathbf{x}}^+$. Such a timing sequence provides the start and end points of each of the keyword phonemes, and can either be provided by manual annotators or localized with a forced alignment algorithm, as discussed in [15]. Let us define the training set as $\mathcal{T}_{\text{train}} \equiv \{(p^{k_i}, \overline{\mathbf{x}}_i^+, \overline{s}_i^+, \overline{\mathbf{x}}_i^-)\}_{i=1}^m$. For each keyword in the training set there is only one positive utterance and one negative utterance,

hence $|X_k^+| = 1$, $|X_k^-| = 1$ and $|\mathcal{K}_{\text{train}}| = m$, and the AUC over the training set becomes

$$A_{\text{train}} = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{f(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^+) > f(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^-)} \ .$$

The selection of a model maximizing this AUC is equivalent to minimizing the loss

$$\mathcal{L}^{0/1}(f) = 1 - A_{\text{train}} = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{f(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^+) > f(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^-)} \ .$$

The loss $\mathcal{L}^{0/1}$ is unfortunately not suitable for model training since it is a combinatorial quantity that is difficult to minimize directly. We instead adopt a strategy commonly used in large margin classifiers and employ the convex hinge-loss function,

$$\mathcal{L}(f) = \frac{1}{m} \sum_{i=1}^{m} \ [1 - f(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^+) + f(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^-)]_+, \tag{1}$$

where $[a]_+$ denotes $\max\{0,\ a\}$. The hinge loss $\mathcal{L}(f)$ upper bounds $\mathcal{L}^{0/1}(f)$: since for any real numbers a and b, $[1-a+b]_+ \geq \mathbb{1}_{a \leq b}$, and moreover, when $\mathcal{L}(f) = 0$, then $A_{\text{train}} = 1$, and for any a and b, $[1-a+b]_+ = 0 \Rightarrow a > b+1 \Rightarrow a > b$. The hinge-loss is related to the ranking loss used in both SVMs for ordinal regression [13] and Ranking SVM [16]. These approaches have shown to be successful over highly unbalanced problems, such as information retrieval [16, 12], using the hinge loss is hence appealing to the keyword spotting problem. The analysis of the relationships between the hinge loss presented in Equation 1 and the generalization performance of our keyword spotter is differed to Section 3.4, where we show that minimizing the hinge loss yields a keyword spotter likely to attain a high AUC over unseen data.

Our keyword spotter f is parameterized as

$$f_{\mathbf{w}}(\overline{\mathbf{x}}, \overline{p}^k) = \max_{\overline{s}} \ \mathbf{w} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}, \overline{p}^k, \overline{s}) \ , \tag{2}$$

where $\mathbf{w} \in \mathbb{R}^n$ is a vector of importance weights, $\boldsymbol{\phi}(\overline{\mathbf{x}}, \overline{p}^k, \overline{s})$ is a feature function vector, measuring different characteristics related to the confidence that the phoneme sequence $\overline{p}^k$ representing the keyword k is uttered in $\overline{\mathbf{x}}$ with the time span $\overline{s}$. Formally, $\boldsymbol{\phi}$ is a function defined as $\boldsymbol{\phi} : \mathcal{X}^* \times (\mathcal{P} \times \mathbb{N})^* \to \mathbb{R}^n$. In this study we used 7 feature function ($n = 7$), which are similar to those employed in [15]. These functions are described only briefly for the sake of completeness. There are four **phoneme transition functions**, which aim at detecting transition between phonemes. For this purpose, they compute the frame distance between the frames before and after a hypothesized transition point. Formally,

$$\forall i = 1, 2, 3, 4, \quad \phi_i(\overline{\mathbf{x}}, \overline{p}^k, \overline{s}) = \frac{1}{L} \sum_{j=2}^{L-1} d(\mathbf{x}_{s_j - i}, \mathbf{x}_{s_j + i}) \ , \tag{3}$$

where d refers to the Euclidean distance and L refers to the number of phonemes in keyword k.
The **frame-based phoneme classifier function** relies on a frame-based phoneme classifier to measure the match between each frame and the hypothesized phoneme class,

$$\phi_5(\overline{\mathbf{x}}, \overline{p}^k, \overline{s}) = \frac{1}{L} \sum_{i=1}^{L} \sum_{t=s_i}^{s_{i+1}-1} \frac{1}{s_{i+1} - s_i} \, g(\mathbf{x}_t, p_i) \tag{4}$$

where $g : \mathcal{X} \times \mathcal{P} \rightarrow \mathbb{R}$ refers to the phoneme classifier, which returns a confidence that the acoustic feature vector at the t-th frame, $\mathbf{x}_t$, represents a specific phoneme p_i. Different phoneme classifiers might be applied for this feature. In our case, we conduct experiments relying on two alternative solutions. The first assessed classifier is the hierarchical large-margin classifier presented in [10], while the second classifier is a Bayes classifier with one Gaussian Mixture per phoneme class. In the first case, g is defined as the phoneme confidence outputted by the classifier, while, in the second case, g is defined as the log posterior of the class $g(\mathbf{x}, p) = \log(P(p|\mathbf{x}))$. The presentation of the training setup, as well as, the empirical comparison of both solutions, are deferred to Section 4.
The **phoneme duration function** measures the adequacy of the hypothesized segmentation $\overline{s}$, with respect to a duration model,

$$\phi_6(\overline{\mathbf{x}}, \overline{p}^k, \overline{s}) = \frac{1}{L} \sum_{i=1}^{L} \log \mathcal{N}(s_{i+1} - s_i; \mu_{p_i}, \sigma^2_{p_i}) \ , \tag{5}$$

where $\mathcal{N}$ denotes the likelihood of a Gaussian duration model, whose mean μ_p and variance σ^2_p parameters for each phoneme p are estimated over the training data.
The **speaking rate function** measures the stability of the speaking rate,

$$\phi_7(\overline{\mathbf{x}}, \overline{p}^k, \overline{s}) = \frac{1}{L} \sum_{i=2}^{L} (r_i - r_{i-1})^2, \tag{6}$$

where r_i denotes the estimate of the speaking rate for the i-th phoneme,

$$r_i = \frac{s_{i+1} - s_i}{\mu_{p_i}}.$$

This set of seven functions has been used in our experiments. Of course, this set can easily be extended to incorporate further features, such as confidences from a triphone frame-based classifier or the output of a more refined duration model.

In other words, our keyword spotter outputs a confidence score by maximizing a weighted sum of feature functions over all possible time-spans. This maximization corresponds to a search over an exponentially large number of time spans. Nevertheless, it can be performed efficiently by selecting decomposable feature functions, which allows the application of dynamic programming techniques, like these used in HMMs [25]. [15] gives a detailed discussion about the efficient computation of Equation 2.

3.3 An Iterative Training Algorithm

In this section we describe an iterative algorithm for finding the weight vector $\mathbf{w}$. We show in the sequel that the weight vector $\mathbf{w}$ found in this processes minimizes the loss $\mathcal{L}(f_{\mathbf{w}})$, hence minimizes the loss $\mathcal{L}^{0/1}$ and in turn resulted with a keyword spotting which attains a high AUC over the training set. We also show that the learned weight vector have good generalization properties on the test set.

The procedure starts by initializing the weight vector to be the zero vector, $\mathbf{w}_0 = 0$. Then, at iteration $i \geq 1$, the algorithm examines the i-th training example $(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^+, \overline{s}_i^+, \overline{\mathbf{x}}_i^-)$. The algorithm first predicts the best time span of the keyword phoneme sequence $\overline{p}^{k_i}$ in the negative utterance $\overline{\mathbf{x}_i^-}$,

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

Then, the algorithm considers the loss on the i-th training example and checks that the difference between the score assigned to the positive utterance and the score assigned to the negative example is greater than 1. Formally, define

$$\Delta\boldsymbol{\phi}_i = \boldsymbol{\phi}(\overline{\mathbf{x}}_i^+, \overline{p}_i^k, \overline{s}_i^+) - \boldsymbol{\phi}(\overline{\mathbf{x}}_i^-, \overline{p}_i^k, \overline{s}_i^-).$$

If $\mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i \geq 1$ the algorithm keeps the weight vector for the next iteration, namely, $\mathbf{w}_i = \mathbf{w}_{i-1}$. Otherwise, the algorithm updates the weight vector to minimizes the following optimization problem

$$\mathbf{w}_i = \arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_{i-1}\|^2 + \ c\ [1 - \mathbf{w} \cdot \Delta\boldsymbol{\phi}_i]_+, \tag{8}$$

where the hyperparameter $c \geq 1$ controls the trade-off between keeping the new weight vector close to the previous one and satisfying the constraint for the current example. Equation (8) can analytically be solved in closed form [9], yielding

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

where

$$\alpha_i = \min\left\{c, \frac{[1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i]_+}{\|\Delta\boldsymbol{\phi}_i\|^2}\right\}. \tag{9}$$

This update is referred to as *passive-aggressive*, since the algorithm *passively* keeps the previous weight ($\mathbf{w}_i = \mathbf{w}_{i-1}$) if the loss of the current training example is already zero ($[1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i]_+ = 0$), while it *aggressively* updates the weight vector to compensate this loss otherwise. At the end of the training procedure, when all training examples have been visited, the best weight $\mathbf{w}$ among $\{\mathbf{w}_0, \ldots \mathbf{w}_m\}$ is selected over a set of validation examples $\mathcal{T}_{\text{valid}}$. The hyperparameter c is also selected to optimize performance on the validation data. The pseudo-code of the algorithm is given in Algorithm 4.

Input: Training set $\mathcal{T}_{\text{train}}$, validation set $\mathcal{T}_{\text{valid}}$; parameter c.
Initialize: $\mathbf{w}_0 = 0$.

Loop: for each $(p^{k_i}, \overline{\mathbf{x}}_i^+, \overline{s}_i^+, \overline{\mathbf{x}}_i^-) \in \mathcal{T}_{\text{train}}$

1. let $\overline{s}_i^- = \arg\max_{\overline{s}} \ \mathbf{w}_{i-1} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}_i^-, \overline{p}^{k_i}, \overline{s})$
2. let $\Delta\boldsymbol{\phi}_i \ = \ \boldsymbol{\phi}(\overline{\mathbf{x}}_i^+, \overline{p}^{k_i}, \overline{s}_i^+) - \boldsymbol{\phi}(\overline{\mathbf{x}}_i^-, \overline{p}^{k_i}, \overline{s}_i^-)$
3. if $\mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i < 1$ then

   let $\alpha_i = \min\left\{c, \dfrac{1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i}{\|\Delta\boldsymbol{\phi}_i\|^2}\right\}$
   update $\mathbf{w}_i = \mathbf{w}_{i-1} + \alpha_i \cdot \Delta\boldsymbol{\phi}_i$

Output: $\mathbf{w}$ achieving the highest AUC over $\mathcal{T}_{\text{valid}}$:

$$\mathbf{w} = \text{argmin}_{\mathbf{w} \in \{\mathbf{w}_1, \ldots, \mathbf{w}_m\}} \frac{1}{m_{\text{val}}} \sum_{j=1}^{m_{\text{val}}} \mathbb{1}_{\max_{\overline{s}^+} \mathbf{w} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}_j^+, \overline{p}^{k_j}, \overline{s}^+) > \max_{\overline{s}^-} \mathbf{w} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}_j^-, \overline{p}^{k_j}, \overline{s}^-)}$$

Figure 4: Passive Aggressive Training

3.4 Analysis

In this section, we derive theoretical bounds on the performance of our keyword spotter. Let us first define the *cumulative AUC* on the training set $\mathcal{T}_{\text{train}}$ as follows

$$\hat{A}_{\text{train}} = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}_{\mathbf{w}_{i-1} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}_i^+, \overline{p}^{k_i}, \overline{s}_i^+) > \mathbf{w}_{i-1} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}_i^-, \overline{p}^{k_i}, \overline{s}_i^-)} \ , \tag{10}$$

where $\overline{s}_i^-$ is generated every iteration step according to (7). The examination of the cumulative AUC is of great interest as it provides an estimator for the generalization performance. Note that at each iteration step the algorithm receives new example $(p^{k_i}, \overline{\mathbf{x}}_i^+, \overline{s}_i^+, \overline{\mathbf{x}}_i^-)$ and predicts the time span of the keyword in the negative instance $\overline{\mathbf{x}}_i^-$ using the previous weight vector $\mathbf{w}_{i-1}$. Only after the prediction is made the algorithm suffers loss by comparing its prediction to the true time span $\overline{s}_i^+$ of the keyword on the positive utterance $\overline{\mathbf{x}}_i^+$. The cumulative AUC is a weighted sum of the performance of the algorithm on the next unseen training example and hence it is a good estimation to the performance of the algorithm on unseen data during training.

Our first theorem states a competitive bound. It compares the cumulative AUC of the weight vectors series, $\{\mathbf{w}_1, \ldots, \mathbf{w}_m\}$, resulted from the iterative algorithm to the best fixed weight vector, $\mathbf{w}^\star$, chosen in hindsight, and essentially proves that, for any sequence of examples, our algorithms cannot do much worse than the best fixed weight vector. Formally, it shows that the cumulative area *above* the curve, $1 - \hat{A}_{\text{train}}$, is smaller than the weighted average loss $\mathcal{L}(f_{\mathbf{w}^\star})$ of the best fixed weight vector $\mathbf{w}^\star$ and its weighted complexity, $\|\mathbf{w}^\star\|$. That is, the

cumulative AUC of the iterative training algorithm is going to be high, given that the loss of the best solution is small, the complexity of the best solution is small and that the number of training examples, m, is sufficiently large.

Theorem 1. *Let $\mathcal{T}_{train} = \{(\overline{p}^{k_i}, \overline{\mathbf{x}}_i^+, \overline{s}_i^+, \overline{\mathbf{x}}_i^-)\}_{i=1}^m$ be a set of training examples and assume that for all k, $\overline{\mathbf{x}}$ and $\overline{s}$ we have that $\|\boldsymbol{\phi}(\overline{\mathbf{x}}, \overline{p}^k, \overline{s})\| \leq 1/\sqrt{2}$. Let $\mathbf{w}^\star$ be the best weight vector selected under some optimization criterion by observing all instances in hindsight. Let $\mathbf{w}_1, \ldots, \mathbf{w}_m$ be the sequence of weight vectors obtained by the algorithm in Algorithm 4 given the training set $\mathcal{T}_{train}$. Then,*

$$1 - \hat{A}_{\text{train}} \leq \frac{1}{m}\|\mathbf{w}^\star\|^2 + \frac{2c}{m}\mathcal{L}(f_{\mathbf{w}^\star}) \tag{11}$$

where $c \geq 1$ and $\hat{A}_{\text{train}}$ is the cumulative AUC defined in Equation 10.

Proof. Denote by $\ell_i(\mathbf{w})$ the instantaneous loss the weight vector $\mathbf{w}$ suffers on the i-th example, that is,

$$\ell_i(\mathbf{w}) = [1 - \mathbf{w} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}_i^+, \overline{p}^{k_i}, \overline{s}_i^+) + \max_{\overline{s}} \ \mathbf{w} \cdot \boldsymbol{\phi}(\overline{\mathbf{x}}_i^-, \overline{p}^{k_i}, \overline{s})]_+$$

The proof of the theorem relies on Lemma 1 and Theorem 4 in [9]. Lemma 1 in [9] implies that,

$$\sum_{i=1}^m \alpha_i \left(2\ell_i(\mathbf{w}_{i-1}) - \alpha_i\|\Delta\boldsymbol{\phi}_i\|^2 - 2\ell_i(\mathbf{w}^\star)\right) \leq \|\mathbf{w}^\star\|^2. \tag{12}$$

Now if the algorithm makes a prediction mistake and the predicted confidence of the best time span of the keyword in a negative utterance is higher than the confidence of the true time span of the keyword in the positive example then $\ell_i(\mathbf{w}_{i-1}) \geq 1$. Using the assumption that $\|\boldsymbol{\phi}(\overline{\mathbf{x}}, \overline{p}^k, \overline{s})\| \leq 1/\sqrt{2}$, which means that $\|\Delta\boldsymbol{\phi}(\overline{\mathbf{x}}, \overline{p}^k, \overline{s})\|^2 \leq 1$, and the definition of α_i given in Equation 9, when substituting $[1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i]_+$ for $\ell_i(\mathbf{w}_{i-1})$ in its numerator, we conclude that if a prediction mistake occurs then it holds that

$$\alpha_i\ell_i(\mathbf{w}_{i-1}) \geq \min\left\{\frac{\ell_i(\mathbf{w}_{i-1})}{\|\Delta\boldsymbol{\phi}_i\|^2}, c\right\} \geq \min\{1, c\} = 1. \tag{13}$$

Summing over all the prediction mistakes made on the entire training set $\mathcal{T}_{\text{train}}$ and taking into account that $\alpha_i\ell_i(\mathbf{w}_{i-1})$ is always non-negative, we have

$$\sum_{i=1}^m \alpha_i\ell_i(\mathbf{w}_{i-1}) \geq \sum_{i=1}^m \mathbb{1}_{\mathbf{w}_{i-1}\cdot\boldsymbol{\phi}(\overline{\mathbf{x}}_i^+, \overline{p}^{k_i}, \overline{s}_i^+) \leq \mathbf{w}_{i-1}\cdot\boldsymbol{\phi}(\overline{\mathbf{x}}^-, \overline{p}^{k_i}, \overline{s}_i^-)}. \tag{14}$$

Again using the definition of α_i, we know that $\alpha_i\ell_i(\mathbf{w}^\star) \leq c\ell_i(\mathbf{w}^\star)$ and that $\alpha_i\|\Delta\boldsymbol{\phi}_i\|^2 \leq \ell_i(\mathbf{w}_{i-1})$. Plugging these two inequalities and (14) into (12) we get

$$\sum_{i=1}^m \mathbb{1}_{\mathbf{w}_{i-1}\cdot\boldsymbol{\phi}(\overline{\mathbf{x}}_i^+, \overline{p}^{k_i}, \overline{s}_i^+) \leq \mathbf{w}_{i-1}\cdot\boldsymbol{\phi}(\overline{\mathbf{x}}^-, \overline{p}^{k_i}, \overline{s}_i^-)} \leq \|\mathbf{w}^\star\|^2 + 2c\sum_{i=1}^m \ell_i(\mathbf{w}^\star). \tag{15}$$

The theorem follows by replacing the sum over prediction mistakes to a sum over prediction hits and plugging the definition of the cumulative AUC given in (10). □

The next theorem states that the output of our algorithm is likely to have good generalization, namely, 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 Theorem 1. Assume that the training set $\mathcal{T}_{train}$ and the validation set $\mathcal{T}_{valid}$ are both sampled i.i.d. from a distribution $\mathcal{D}$. Denote by m_{valid} the size of the validation set. With probability of at least $1-\delta$ we have*

$$1 - A = \mathbb{E}_{\mathcal{D}}\left[\mathbb{1}_{f(\overline{\mathbf{x}}_i^+,\overline{p}^{k_i}) \le f(\overline{\mathbf{x}}^-,\overline{p}^{k_i})}\right] = \Pr_{\mathcal{D}}\left[f(\overline{\mathbf{x}}_i^+,\overline{p}^{k_i}) \le f(\overline{\mathbf{x}}^-,\overline{p}^{k_i})\right] \quad \le$$
$$\frac{1}{m}\sum_{i=1}^{m}\ell_i(\mathbf{w}^\star) + \frac{\|\mathbf{w}^\star\|^2}{m} + \frac{\sqrt{2\ln(2/\delta)}}{\sqrt{m}} + \frac{\sqrt{2\ln(2m/\delta)}}{\sqrt{m_{\text{valid}}}} \quad , \quad (16)$$

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

$$\ell_i(\mathbf{w}) = [1 - \mathbf{w}\cdot\boldsymbol{\phi}(\overline{\mathbf{x}}_i^+,\overline{p}^{k_i},\overline{s}_i^+) + \max_{\overline{s}}\ \mathbf{w}\cdot\boldsymbol{\phi}(\overline{\mathbf{x}}_i^-,\overline{p}^{k_i},\overline{s})]_+ \ .$$

The proof of the theorem goes along the lines of the proof in [15]. The theorem states that the resulted $\mathbf{w}$ of the iterative algorithm generalizes, with high probability, and is going to have high expected AUC on unseen test data.

4 Experiments and Results

We started by training the iterative algorithm on the TIMIT training set. We then conducted two types of experiments to evaluate the effectiveness of the proposed discriminative method. First, we compared the performance of the discriminative method to a standard monophone HMM keyword spotter on the TIMIT test set. Second, we compared the robustness of both the discriminative method and the monophone HMM with respect to changing recording conditions by using the models trained on the TIMIT, evaluated on the Wall Street Journal (WSJ) corpus.

4.1 The TIMIT Experiments

The TIMIT corpus [11] consists of read speech from 630 American speakers, with 10 utterances per speaker. The corpus provides manually aligned phoneme and word transcriptions for each utterance. It also provides a standard split into training and test data. From the training part of the corpus, we extracted three disjoint sets consisting of 1500, 300 and 200 utterances. The first set was used

Model	AUC
HMM/Viterbi	0.942
HMM/Ratio	0.952
Discriminative/GMM	0.971
Discriminative/Hier	0.996

Table 1: AUC of different models trained on the TIMIT training set and evaluated on the TIMIT test set (the higher the better)

as the training set of the phoneme classifier and was used by our fifth feature function ϕ_5. The second set was used as the training set for our discriminative keyword spotter, while the third set was used as the validation set to select the hyperparameter c and the best weight vector $\mathbf{w}$ seen during training. The test set was solely used for evaluation purposes. From each of the last two splits of the training set, 200 words of length greater than or equal to 4 phonemes were chosen in random. From the test set 80 words were chosen in random as described below.

Mel Frequency Cepstral Coefficients (MFCC), along with their first (Δ) and second derivatives ($\Delta\Delta$), were extracted every 10 ms. These features were used by the first five feature functions $\phi_1, \ldots, \phi_5$. Two types of phoneme classifiers were used for the fifth feature function ϕ_5, namely, a large margin phoneme classifier [10] and a GMM model. Both classifiers were trained to predict 39 phoneme classes [22] over the first part of the training set. The large margin classifier corresponds to a hierarchical classifier with Gaussian kernel, as presented in [10], where the score assigned to each frame for a given phoneme was used as the function g in Equation (4). The GMM model corresponded to a Bayes classifier combining one GMM per class and the phoneme prior probabilities, both learned from the training data. In that case, the log posterior of a phoneme given the frame vector was used as the function g in Equation (4). The hyperparameters of both phoneme classifiers were selected to maximize the frame accuracy over part of the training data held out during parameter fitting. In the following, the discriminative keyword spotter relying on the features from the hierarchical phoneme classifier is referred to as *Discriminative/Hier*, while the model relying on the GMM log posteriors is referred to as *Discriminative/GMM*.

We compared the results of both *Discriminative/Hier* and *Discriminative/GMM* to a monophone HMM baseline, in which each phoneme were modeled with a left-right HMM of 5 emitting states. The density of each state was modeled with a 40-Gaussian GMM. Training was performed over the whole TIMIT training set. *Embedded training* was applied, i.e., after an initial training phase relying on the provided phoneme alignment, a second training phase which dynamically determines the most likely alignment was applied. The hyperparameters of this model (the number of states per phoneme, the number of Gaussians per state,

as well as the number of Expectation-Maximization iterations) were selected to maximize the likelihood of an held-out validation set.

The phoneme models of the trained HMM were then used to build a keyword spotting HMM, composed of two sub-models: the keyword model and the garbage model, as illustrated on Figure 1. 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 all phoneme HMMs fully connected to each other, which estimated the likelihood of any phoneme sequence. The overall HMM fully connected the keyword model and the garbage model. The detection of a keyword in a given utterance was performed by checking whether the Viterbi best path passes through the keyword model, as explained in Section 2. In this model, the keyword transition probability set the trade-off between the true positive rate and the ROC curve was plotted by varying this probability. This model is referred to as *HMM/Viterbi*.

We also experimented an alternative decoding strategy, in which the system output the ratio of the likelihood of the acoustic sequence knowing the keyword was uttered versus the likelihood of the sequence knowing the keyword was *not* uttered, as discussed in Section 2. In this case, the first likelihood was determined by an HMM forcing an occurence of the keyword, and the second likelihood was determined by the garbage model, as illustrated on Figure 2. This likelihood-ratio strategy is referred to as *HMM/Ratio* in the following.

The evaluation of discriminative and HMM-based models was performed over 80 keywords, randomly selected among the words occurring in the TIMIT test set. This random sampling of the keyword set aimed at evaluating the expected performance over any keyword. For each keyword k, we considered a spotting problem, which consisted of a set of positive utterances X_k^+ and a set of negative utterance X_k^-. Each positive set X_k^+ contained between 1 and 20 sequences, depending on the number of occurrences of k in the TIMIT test set. Each negative set contained 20 sequences, randomly sampled among the utterances of TIMIT which does not contain k. This setup represented an unbalanced problem, with only 10% of the sequences being labeled as positive.

Table 1 reports the average AUC results of the 80 test keywords, for different models trained on the TIMIT training set and evaluated on the TIMIT test set. These results show the advantage of our discriminative approach. The two discriminative models outperforms the two HMM-based models. The improvement introduced by our discriminative model algorithm can be observed when comparing the performance of *Discriminative/GMM* to the performance of the HMM spotters. In that case, both spotters rely on GMMs to estimate the frame likelihood given a phoneme class. In our case we use that probability to compute the feature ϕ_5, while the HMM uses it as the state emission probability.

Moreover, our keyword spotter can benefit from non-probabilistic frame-based classifiers, as illustrated with *Discriminative/Hier*. This model relies on the output of a large margin classifier, which outperforms all other models, and reaches a mean AUC of 0.996. In order to verify whether the differences observed on averaged AUC could be due only to a few keywords, we applied the Wilcoxon

Best Model	Keywords
Discriminative/Hier	absolute admitted apartments apparently argued controlled depicts dominant drunk efficient followed freedom introduced millionaires needed obvious radiation rejected spilled street superb sympathetically weekday **(23 keywords)**
HMM/Ratio	materials **(1 keyword)**
No differences	aligning anxiety bedrooms brand camera characters cleaning climates creeping crossings crushed decaying demands dressy episode everything excellent experience family firing forgiveness fulfillment functional grazing henceforth ignored illnesses imitate increasing inevitable January mutineer package paramagnetic patiently pleasant possessed pressure recriminations redecorating secularist shampooed solid spreader story strained streamlined stripped stupid surface swimming unenthusiastic unlined urethane usual walking **(56 keywords)**

Table 2: The distribution of the 80 keywords among the models which better spotted them. Each row in the table represents the keywords for which the model written at the beginning of the row received the highest AUC. The models were trained on the TIMIT training set and evaluated on the TIMIT test set.

test [26] to compare the results of both HMM approaches (*HMM/Viterbi* and *HMM/Ratio*) with the results of both discriminative approaches (*Discriminative/GMM* and *Discriminative/Hier*). At the 90% confidence level, the test rejected this hypothesis, showing that the performance gained of the discriminative approach is consistent over over the keyword set.

Table 2 further presents the performance per keyword and compares the results of the best HMM configuration, *HMM/Ratio* to the performance of the best discriminative configuration, *Discriminative/Hier*. Out of total 80 keywords, 23 keywords were better spotted with the discriminative model, 1 keyword was better spotted with the HMM, and both models yielded the same spotting accuracy for 56 keywords. The discriminative model seems to be better for shorter keywords, as it outperforms the HMM for most of the keywords of 5 phonemes or less (e.g., *drunk*, *spilled*, *street*).

4.2 The WSJ Experiments

WSJ [24] is a large corpus of American English. It consists in read and spontaneous speech corresponding to the reading and the dictation of articles from the Wall Street Journal. In the following, all models were trained on the TIMIT training set and evaluated on the `si_tr_s` subset of WSJ. This subset corresponds to the recordings of 200 speakers. Compared to TIMIT, this subset introduce several variations, both regarding the type of sentences recorded and

Model	AUC
HMM/Viterbi	0.868
HMM/Ratio	0.884
Discriminative/GMM	0.922
Discriminative/Hier	0.914

Table 3: AUC of different models trained on the TIMIT training set and evaluated on the `si_tr_s` subset of WSJ (the higher the better)

the recording conditions [24]. These experiments hence evaluate the robustness of the different approaches when they encounter differing conditions for training and testing. Like for TIMIT, the evaluation is performed over 80 keywords randomly selected from the corpus transcription. For each keyword k, the evaluation was performed over a set X_k^+, containing between 1 and 20 positive sequences, and a X_k^-, containing 20 randomly selected negative sequences. This setup also represents an unbalanced problem, with 27% of the sequences being labeled as positive.

Table 3 reports the average AUC results of the 80 test keywords, for different models trained on the TIMIT training set and evaluated on the `si_tr_s` subset of WSJ. Overall, the results show that the differences between the TIMIT training conditions and the WSJ test conditions affect the performance of all models. However, the measured performance still yield acceptable performance in all cases (AUC of 0.868 in the worse case). Comparing the individual model performance, the WSJ results confirm the conclusions of TIMIT experiments and the discriminative spotters outperform the HMM-based alternatives. For the HMM models, *HMM/Ratio* outperforms *HMM/Viterbi* like in the TIMIT experiments. For the discriminative spotters, *Discriminative/GMM* outperforms *Discriminative/Hier*, which was not the case over TIMIT. Since these two models only differ in the frame-based classifier used as the the feature function ϕ_5, this result certainly indicates that the hierarchical frame-based classifier on which *Discriminative/Hier* relies is less robust to the acoustic condition changes than the GMM alternative. Like for TIMIT, we checked whether the differences observed on the whole set could be due to a few keywords. The Wilcoxon test rejected this hypothesis at the 90% confidence level, for the 4 tests comparing *Discriminative/GMM* and *Discriminative/Hier* to *HMM/Viterbi* and *HMM/Hier*.

We further compared the best discriminative spotter, *Discriminative/GMM*, and the best HMM spotter *HMM/Ratio* over each keyword. These results are summarized in Table 4. Out of the 80 keywords, the discriminative model outperforms the HMM for 50 keywords, the HMM outperforms the discriminative model for 20 keywords and both models yield the same results for 10 keywords. Like for the TIMIT experiments, the discriminative model is shown to be especially advantageous for short keywords, with 5 phonemes or less (e.g., *Adams*,

Best Model	Keywords
Discriminative/Hier	Adams additions Allen Amerongen apiece buses Bushby Colombians consistently cracked dictate drop fantasy fills gross Higa historic implied interact kings list lobby lucrative measures Melbourne millions Munich nightly observance owning plus proudly queasy regency retooling Rubin scramble Seidler serving significance sluggish strengthening Sutton's tariffs Timberland today truths understands withhold Witter's **(50 keywords)**
HMM/Ratio	artificially Colorado elements Fulton itinerary longer lunchroom merchant mission multilateral narrowed outlets Owens piper replaced reward sabotaged shards spurt therefore **(20 keywords)**
No differences	aftershocks Americas farms Flamson hammer homosexual philosophically purchasers sinking steel-makers **(10 keywords)**

Table 4: The distribution of the 80 keywords among the models which better spotted them. Each row in the table represents the keywords for which the model written at the beggining of the row received the highest AUC. The models were trained on the TIMIT training set but evaluated on the `si_tr_s` subset of WSJ

kings, *serving*).

Overall, the experiments over both WSJ and TIMIT highlight the advantage of our discriminative learning method.

5 Conclusions

This chapter introduces a discriminative method to the keyword spotting problem. In this task, the model receives a keyword and a spoken utterance as input and should decide whether the keyword is uttered in the utterance. Keyword spotting corresponds to an unbalanced detection problem, since, in standard setups, most of tested utterances do not contain the targeted keyword. In that unbalanced context, the AUC is generally used for evaluation. This work proposed a learning algorithm, which aims at maximizing the AUC over a set of training spotting problems. Our strategy is based on a large margin formulation of the task, and relies on an efficient iterative training procedure. The resulting model contrasts with standard approaches based on HMMs, for which the training procedure does not rely on a loss directly related to the spotting task. Compared to such alternatives, our model is shown to yield significant improvements over various spotting problems on the TIMIT and the WSJ corpus. For instance, the best HMM configuration over TIMIT reaches AUC of 0.953, compared to AUC of 0.996 for the best discriminative spotter.

Several potential directions of research can be identified from this work. In its current configuration, our keyword spotter relies on the output of a pre-

trained frame-based phoneme classifier. It would be of a great interest to learn the frame-based classifier and the keyword spotter jointly, so that all model parameters are selected to maximize the performance on the final spotting task.

Also, our work currently represents keywords as sequence of phonemes, without considering the neighboring context. Possible improvement might results from the use of phonemes in context, such as triphones. We hence plan to investigate the use of triphones in a discriminative framework, and to compare the resulting model to triphone-based HMMs. More generally, our model parameterization offers greater flexibility to incorporate new features, compared to probabilistic approaches such as HMMs. Therefore, in addition to triphones, features extracted from the speaker identity, the channel characteristics or the linguistic context could possibly be included to improve performance.

Acknowledgments

This research was partly performed while David Grangier was visiting Google Inc. (Mountain View, USA), and while Samy Bengio was with the IDIAP Research Institute (Martigny, Switzerland). This research was supported by the European PASCAL Network of Excellence and the DIRAC project.

References

[1] L. R. Bahl, P. F. Brown, P. de Souza, and R. L. Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1986.

[2] Y. Benayed, D. Fohr, J. P. Haton, and G. Chollet. Confidence measures for keyword spotting using support vector machines. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 2003.

[3] 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*, 2004.

[4] J. A. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden Markov models. Technical Report TR-97-021, International Computer Science Institute, Berkeley, CA, USA, 1998.

[5] J. M. Boite, H. Bourlard, B. D'hoore, and M. Haesen. Keyword recognition using template concatenation. In *Proceedings of the European Conference on Speech and Communication Technologies (EUROSPEECH)*. International Speech Communication Association, 1993.

[6] J. S. Bridle. An efficient elastic-template method for detecting given words in running speech. In *Proceedings of the British Acoustic Society Meeting*. British Acoustic Society, 1973.

[7] E. Chang. *Improving Word Spotting Performance with Limited Training Data*. PhD thesis, Massachusetts Institute of Technology (MIT), 1995.

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

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

[10] 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.

[11] J. S. Garofolo. TIMIT acoustic-phonetic continuous speech corpus. Technical Report LDC93S1, Linguistic Data Consortium, Philadelphia, PA, USA, 1993.

[12] D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI).*, 2008.

[13] R. Herbrich, T. Graepel, and K. Obermayer. Large marging rank boundaries for ordinal regression. In A. Smola, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*. MIT Press, 2000.

[14] A. L. Higgins and R. E. Wohlford. Keyword recognition using template concatenation. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1985.

[15] Y. Singer J. Keshet, S. Shalev-Shwartz and D. Chazan. A large margin algorithm for forced alignment. In J. Keshet and S. Bengio, editors, *Automatic Speech and Speaker Recognition: Large Margin and Kernel Methods*. Wiley, 2009.

[16] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*. Association for Computing Machinery, 2002.

[17] J. Junkawitsch, G. Ruske, and H. Hoege. Efficient methods in detecting keywords in continuous speech. In *Proceedings of the European Conference on Speech and Communication Technologies (EUROSPEECH)*. International Speech Communication Association, 1997.

[18] T. Kawabata, T. Hanazawa, and K. Shikano. Word spotting method based on hmm phoneme recognition. *Journal of the Acoustical Society of America (JASA)*, 1(84), 1988.

[19] J. Keshet and S. Bengio. Introduction. In J. Keshet and S. Bengio, editors, *Automatic Speech and Speaker Recognition: Large Margin and Kernel Methods*. Wiley, 2009.

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

[21] K. F. Lee and H. F. Hon. Large-vocabulary speaker-independent continuous speech recognition using HMM. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1988.

[22] K. F. Lee and H. W. Hon. Speaker independent phone recognition using hidden Markov models. *Transactions on Acoustics, Speech and Signal Processing (TASSP)*, 11(37), 1989.

[23] H.B. Mann and D.R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Annals of Mathematical Statistics*, 18(1), 1947.

[24] D.B. Paul and J.M. Baker. The design for the Wall Street Journal-based CSR corpus. In *Proceedings of the Human Language Technology Conference (HLT)*. Morgan Kaufmann, 1992.

[25] L. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Upper Saddle River, NJ, USA, 1993.

[26] J.A. Rice. *Rice, Mathematical Statistics and Data Analysis*. Duxbury Press, 1995.

[27] R. C. Rose and D. B. Paul. A hidden Markov model based keyword recognition system. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1990.

[28] E. D. Sandness and I. Lee Hetherington. Keyword-based discriminative training of acoustic models. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*. IEEE Computer Society, 2000.

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

[30] R. A. Sukkar, A. R. Seltur, M. G. Rahim, and C. H. Lee. Utterance verification of keyword strings using word-based minimum verification error training. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1996.

[31] M. Weintraub. Keyword spotting using SRI's DECIPHER large vocabulary speech recognition system. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1993.

[32] M. Weintraub. LVCSR log-likelihood ratio scoring for keyword spotting. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1995.

[33] M. Weintraub, F. Beaufays, Z. Rivlin, Y. Konig, and A. Stolcke. Neural-network based measures of confidence for word recognition. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE Computer Society, 1997.

[34] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1, 1945.

[35] J. G. Wilpon, L. R. Rabiner, C. H. Lee, and E. R. Goldman. Automatic recognition of keywords in unconstrained speech using hidden Markov models. *Transactions on Acoustics, Speech and Signal Processing (TASSP)*, 38(11), 1990.