Spoken Term Detection Automatically Adjusted for a Given Threshold

Tzeviya Fuchs and Joseph Keshet

***Abstract*—Spoken term detection (STD) is the task of determining whether and where a given word or phrase appears in a given segment of speech. Algorithms for STD are often aimed at maximizing the gap between the scores of positive and negative examples. As such they are focused on ensuring that utterances where the term appears are ranked higher than utterances where the term does not appear. However, they do not determine a detection threshold between the two. In this paper, we propose a new approach for setting an absolute detection threshold for all terms by introducing a new calibrated loss function. The advantage of minimizing this loss function during training is that it aims at maximizing not only the relative ranking scores, but also adjusts the system to use a fixed threshold and thus maximizes the detection accuracy rates. We use the new loss function in the structured prediction setting and extend the discriminative keyword spotting algorithm for learning the spoken term detector with a single threshold for all terms. We further demonstrate the effectiveness of the new loss function by training a deep neural Siamese network in a weakly supervised setting for template-based spoken term detection, again with a single fixed threshold. Experiments with the TIMIT, WSJ and Switchboard corpora showed that our approach not only improved the accuracy rates when a fixed threshold was used but also obtained higher Area Under Curve (AUC).**

***Index Terms*—Spoken term detection, keyword spotting, AUC maximization, structured prediction, deep-neural networks**

I. Introduction

SPOKEN term detection (STD) refers to the proper detection of any occurrence of a given word or phrase in a speech signal. Typically, any such system assigns a confidence score to every term it presumably detects. A speech signal is called *positive* or *negative*, depending on whether or not it contains the desired term. Ideally, an STD system assigns a positive speech input with a score higher than the score it assigns to a negative speech input.

During inference, a detection threshold is chosen to determine the point from which a score would be considered positive or negative. The choice of the threshold represents a trade-off between different operational settings, as a high value of the threshold could cause an excessive amount of false negatives (instances incorrectly classified as negative), whereas a low value of the threshold could cause additional false positives (instances incorrectly classified as positive).

The performance of STD systems can be measured by the Receiver Operation Characteristics (ROC) curve, that is, a plot of the true positive (spotting a term correctly) rate as a function of the false positive (mis-spotting a term) rate. Every point on the graph corresponds to a specific threshold value. The area under the ROC curve (AUC) is the expected performance of the system for all threshold values.

A common practice for finding the threshold is to empirically select the desired value using a cross validation procedure. In [1], the threshold was selected using the ROC curve. Similarly, in [2], [3] and the references therein, the threshold was chosen such that the system maximized the Actual Term Weighted Value (ATWV) score [4]. Additionally, [5] claims that a global threshold that was chosen for all terms was inferior to using a term specific threshold [6].

Recently, two works proposed to normalize the detection scores such that a global threshold could be used for all terms while aiming to maximize ATWV. Karakos *et al.* (2013) [4] suggested to calibrate the scores so that all the keywords in a keyword spotting system would be comparable to each other using a machine learning technique. Vinyals and Wegmann (2014) [7], on the other hand, proposed a surrogate loss function which approximates ATWV by smoothing it with a sigmoid function, and set a fixed threshold of 0.5 for all keywords.

In this paper we propose a new method to embed an automatic adjustment of the detection threshold within a learning algorithm, so that it is fixed and known for all terms. We present two algorithmic implementations of our method: the first is a structured prediction model that is a variant of the discriminative keyword spotting algorithm proposed by [8], [9], [10], and the second implementation is a variant of whole-word Siamese deep network models [11], [12], [13]. Both of these approaches in their original form aim to assign positive speech inputs with higher scores than those assigned to negative speech inputs, and were shown to have good results on several datasets. However, maximizing the gap between the scores of the positive and negative examples only ensures the correct relative order between those examples, and does not fix a threshold between them; therefore it cannot guarantee a correct detection for a global threshold. Our goal is to train a system adjusted to use a global threshold valid for all terms.

In this work, we set the threshold to be a fixed value, and adjust the decoding function accordingly. To do so, we propose a new loss function that trains the ranking function to separate the positive and negative instances; that is, instead of merely assigning a higher score to the positive examples, it rather fixes the threshold to be a certain constant, and assigns the positive examples with scores greater than the threshold, and the negative examples with scores less than the threshold. Additionally, this loss function is a surrogate loss function which extends the hinge loss to penalize misdetected

T. Fuchs and J. Keshet are with the Department of Computer Science, Bar-Ilan University, Ramat-Gan, Israel, Emails: fuchstz@cs.biu.ac.il, jkeshet@cs.biu.ac.il

instances. The new loss function is an upper bound to the ranking loss function, hence minimizing the new loss function can lead to minimization of ranking errors, or equivalently to the maximization of the AUC.

This paper is organized as follows. In Sec. II we formally introduce the spoken term detection problem, and in Sec. III we derive the method with which we restrict the detection threshold to be fixed. We then present the two implementations of our derived method: in Sec. IV we use fully supervised learning to learn the parameters of a structured prediction model, and in Sec. V we use weakly supervised learning to learn the parameters of a deep network. In both of these algorithms, our method sets a detection threshold and maximizes accuracy rates. Next, we present experimental results of both implementations in Sec. VI. Finally, concluding remarks and future directions are discussed in Sec. VII.

II. Problem Setting

In the STD task, we are provided with a speech utterance and a *term* and the goal is to decide whether or not the term is uttered. The term can be provided as a sequence of phonemes or by an acoustic representation given by a speech segment in which the term is known to be uttered.

Throughout the paper, scalars are denoted using lower case Latin letters, e.g., x, and vectors using bold face letters, e.g., $\mathbf{x}$. A sequence of elements is denoted with a bar ($\bar{\mathbf{x}}$) and its length is written as $|\bar{\mathbf{x}}|$.

Formally, the speech signal is represented by a sequence of acoustic feature vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, where each feature vector is d dimensional $\mathbf{x}_t \in \mathbb{R}^d$ for all $1 \le t \le T$. Note that in our setting the number of frames T is not fixed. We denote by $\mathcal{X} = (\mathbb{R}^d)^*$ the set of all finite length sequences over $\mathbb{R}^d$. A sequence of L^r phonemes of a term r is denoted as $\bar{p}^r = (p_1, \ldots, p_{L^r})$, where $p_l \in \mathcal{P}$ for all $1 \le l \le L^r$ and $\mathcal{P}$ is the set of phoneme symbols. We denote by $\mathcal{P}^*$ the set of all finite length sequences over $\mathcal{P}$.

A term is a word or a short phrase and is presented to the system either as a sequence of phonemes in the strongly supervised setting or as an acoustic segment containing the term in the weakly supervised setting. We denote the abstract domain of the term representations (as either a phoneme sequence or an acoustic segment) by $\mathcal{R}$. Our goal is to find a *spoken term detector*, which takes as input a speech segment and a term and returns a binary output indicating whether the term was pronounced in the acoustic segment or not. Most often the spoken term detector is a function that returns a real value expressing the confidence that the target term has been uttered. The confidence score outputted by this function is compared to a *threshold*, and if the score is above the threshold the term is declared to have been pronounced in the speech segment. Formally, the detector is a function f from $\mathcal{X} \times \mathcal{R}$ to $\mathbb{R}$. The detection threshold is denoted by the scalar $\theta \in \mathbb{R}$. Usually there is no single threshold for all terms, and it needs to be adjusted after decoding.

Our goal in this work is to propose a new method to learn the spoken term detector from a training set of examples, so that the model is adjusted to use a fixed given threshold for all terms. The function f is found from a training set of examples, where each example is composed of two speech segments and a representation of a term. Although the training set contains many different terms, the function f should be able to detect any term, not only those already seen in the training phase.

III. Loss function for detection with a fixed threshold

In this section we describe our main idea, whereas in the next sections we propose two implementations: one with a structured prediction model where the training data is fully supervised and the term is given as a phoneme sequence, and the other with a deep learning model where the training data is weakly supervised and the term is given using a segment of speech.

Recall that during inference the input to the detector is a speech segment and a term and the output is a confidence that the term was pronounced in the speech segment, which is compared to a threshold. Since the detection threshold is typically not fixed and does depend on the input term, it is often desired to learn the function f such that the confidence of a speech segment that contains the term is higher than the confidence of a speech segment that does not contain the term.

Formally, let us consider two sets of speech segments. Denote by $\mathcal{X}^{r+}$ a set of speech segments in which the term r is articulated. Similarly, denote by $\mathcal{X}^{r-}$ a set of speech segments in which the term r is not articulated. We assume that term r, and two instances $\bar{\mathbf{x}}^+ \in \mathcal{X}^{r+}$ and $\bar{\mathbf{x}}^- \in \mathcal{X}^{r-}$ are drawn from a fixed but unknown probability distribution, and we denote by $\mathbb{P}\{\pi\}$ and $\mathbb{E}[\pi]$ the probability and the expectation of an event π under this distribution. The probability that the confidence of $\bar{\mathbf{x}}^+$ is higher than the confidence of $\bar{\mathbf{x}}^-$ is the area under the ROC curve (AUC) [14], [15]:

$$AUC = \mathbb{P}\left\{f(\bar{\mathbf{x}}^+, r) > f(\bar{\mathbf{x}}^-, r)\right\}. \tag{1}$$

Instead of keeping a threshold for each term, we adjust f so that the detection threshold will be fixed, and set to a predefined value. Assume that the predefined threshold is θ, then the accuracy in the prediction can be measured by

$$Acc_\theta = \mathbb{P}\left\{f(\bar{\mathbf{x}}^+, r) > \theta \quad \wedge \quad f(\bar{\mathbf{x}}^-, r) < \theta\right\}, \tag{2}$$

where $\wedge$ is the logical conjunction symbol. Hence our goal is to find the parameters of the function f so as to maximize the accuracy Acc_θ for a given threshold θ. Equivalently we find the parameters of function f that minimize the error defined as

$$\begin{aligned} Err_\theta &= 1 - Acc_\theta && (3)\\ &= \mathbb{P}\left\{f(\bar{\mathbf{x}}^+, r) < \theta \quad \vee \quad f(\bar{\mathbf{x}}^-, r) > \theta\right\} && (4)\\ &= \mathbb{E}\left[\mathbb{I}\{f(\bar{\mathbf{x}}^+, r) < \theta\} + \mathbb{I}\{f(\bar{\mathbf{x}}^-, r) > \theta\}\right], && (5)\end{aligned}$$

where $\vee$ is the logical disjunction symbol, and $\mathbb{I}\{\pi\}$ is the indicator function, that equals 1 if the predicate π holds true and 0 otherwise.

Unfortunately, we cannot minimize the error function (5) directly, since it is a combinatorial quantity. A common practice is to replace the error function with a surrogate loss function which is easy to minimize. We suggest to minimize a convex upper-bound to the error function. Specifically, we replace the last term with the hinge upper bound,

$$Err_\theta \leq \mathbb{E}\left[[1+\theta-f(\bar{\mathbf{x}}^+,r)]_+ + [1-\theta+f(\bar{\mathbf{x}}^-,r)]_+\right], \quad (6)$$

where $[\pi]_+ = \max\{\pi, 0\}$. The last upper bound holds true since $\mathbb{I}\{\pi < 0\} \leq [1-\pi]_+$. Adding the margin of 1 means that the function f faces a harder problem: not only does it need to have a confidence greater than θ for a positive speech segment and a confidence lower than θ for a negative speech segment – the confidence value must be at least $\theta+1$ and at most $\theta-1$ for positive and negative speech segments, respectively.

We now turn to present two algorithmic implementations that are aimed at minimizing the loss function derived from (6), namely,

$$\begin{aligned}\ell(\bar{\mathbf{x}}^+,\bar{\mathbf{x}}^-,r;\theta) = {} & [1+\theta-f(\bar{\mathbf{x}}^+,r)]_+ \\ & + \quad [1-\theta+f(\bar{\mathbf{x}}^-,r)]_+. \quad (7)\end{aligned}$$

Hopefully the minimization of this loss function will lead to the minimization of Err_θ in (6).

IV. Structured prediction model

Our first construction is based on previous work on discriminative keyword spotting and spoken term detection [8], [9], [10], where the goal was to maximize the AUC. In this setting we assume that the term is expressed as a sequence of phonemes denoted $\bar{p}^r \in \mathcal{P}^*$.

In this fully-supervised setting we define the *alignment* between a phoneme sequence and a speech signal. We denote by $y_l \in \mathbb{N}$ the start time of phoneme p_l (in frame units), and by $e_l = y_{l+1} - 1$ the end time of phoneme p_l, except for the phoneme p_L, where the end frame is e_L. The alignment sequence $\bar{y}^r$ corresponding to the phonemes sequence $\bar{p}^r$ is a sequence of start-times and an end-time, $\bar{y}^r = (y_1, \ldots, y_L, e_L)$, where y_l is the start-time of phoneme p_l and e_L is the end-time of the last phoneme p_L.

Similar to previous work [8], [9], [10], our detection function f is composed of a predefined set of n *feature functions*, $\{\phi_j\}_{j=1}^n$, each of the form $\phi_j : \mathcal{X}^* \times \mathcal{P}^* \times \mathbb{N}^* \to \mathbb{R}$. Each feature function takes as input an acoustic representation of a speech utterance $\bar{\mathbf{x}} \in \mathcal{X}^*$, together with the term phoneme sequence $\bar{p}^r \in \mathcal{P}^*$, and a candidate alignment sequence $\bar{y}^r \in \mathbb{N}^*$, and returns a scalar in $\mathbb{R}$ which represents the confidence in the suggested alignment sequence given the term r. 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. Our basic set of feature functions is the same as the set used in [8].

We believe that the threshold value for each term depends on the term's phonetic content and its relative duration. In order to allow f to learn these subtle differences from the data we introduced an additional set of 4 feature functions: a feature function representing a bias; a feature function that counts the number of occurrences of a phoneme in a term, i.e., $|\{q|q \in \bar{p}^r\}|$; a feature function holding the number of phonemes in the term, i.e., $|\bar{p}^r|$; and a feature function holding the average length of the phonemes in a term, i.e., $\frac{1}{L^r}\sum_{i=1}^{L^r}(y_{i+1}-y_i)$.

As mentioned above, our goal is to learn a spoken term detector f, which takes as input a sequence of acoustic features $\bar{\mathbf{x}}$, a term $\bar{p}^r$, and returns a confidence value in $\mathbb{R}$. The form of the function f we use is

$$f(\bar{\mathbf{x}},\bar{p}^r) = \max_{\bar{y}} \; \mathbf{w}\cdot\boldsymbol{\phi}(\bar{\mathbf{x}},\bar{p}^r,\bar{y}) \;, \quad (8)$$

where $\mathbf{w} \in \mathbb{R}^n$ is a vector of importance weights 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 term in the utterance by maximizing a weighted sum of the scores returned by the feature function elements over all possible alignment sequences. If the confidence of the function f is above the threshold θ then we predict that the term is pronounced in the signal and located in the time span defined by the alignment sequence $\bar{y}$ that maximizes (8):

$$\bar{y}' = \arg\max_{\bar{y}} \; \mathbf{w}\cdot\boldsymbol{\phi}(\bar{\mathbf{x}},\bar{p}^r,\bar{y}) \;, \quad (9)$$

where the search for the best sequence is practically performed using the Viterbi algorithm as described in [8]. Specifically, the algorithm finds the optimal time segment for the keyword r in the speech signal $\bar{\mathbf{x}}$, and then aligns the phoneme sequence $\bar{p}^r$ within the chosen time segment.

The parameters of the model $\mathbf{w}$ are found by minimizing the loss function defined in (7). In the fully supervised case we use a slightly modified version of it, which is defined as

$$\begin{aligned}&\ell(\mathbf{w},(\bar{\mathbf{x}}^+,\bar{\mathbf{x}}^-,\bar{p}^r,\bar{y}^r);\theta) \\ &\qquad = \left[1+\theta-\mathbf{w}^\top\phi(\bar{\mathbf{x}}^+,\bar{p}^r,\bar{y}^r)\right]_+ \\ &\qquad + \quad \left[1-\theta+\max_{\bar{y}'}\mathbf{w}^\top\phi(\bar{\mathbf{x}}^-,\bar{p}^r,\bar{y}')\right]_+. \quad (10)\end{aligned}$$

This is a convex function in the vector of the parameters $\mathbf{w}$. We use the Passive-Aggressive (PA) algorithm [16], [8] to find the parameters $\mathbf{w}$. The algorithm receives as input a set of training examples $S = \{(\bar{p}^{r_i},\bar{\mathbf{x}}_i^+,\bar{\mathbf{x}}_i^-,\bar{y}^{r_i})\}_{i=1}^m$ 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 $(\bar{p}^{r_i},\bar{\mathbf{x}}_i^+,\bar{\mathbf{x}}_i^-,\bar{y}^{r_i})$ as we now describe.

Denote by $\mathbf{w}_{i-1}$ the value of the weight vector before the ith iteration. We set the next weight vector $\mathbf{w}_i$ to be the minimizer of the following optimization problem,

$$\begin{aligned}\mathbf{w}_i = \operatorname*{argmin}_{\mathbf{w}\in\mathbb{R}^n,\xi\geq 0} \quad & \frac{1}{2}||\mathbf{w}-\mathbf{w}_{i-1}||^2 + C\xi \quad (11)\\ \text{s.t.} \quad & \ell(\mathbf{w}_i,(\bar{p}^{r_i},\bar{\mathbf{x}}_i^+,\bar{\mathbf{x}}_i^-,\bar{y}^{r_i});\theta) \leq \xi \;,\end{aligned}$$

where C serves as a complexity-accuracy trade-off parameter 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 our previous weight vector $\mathbf{w}_{i-1}$. The constraint makes the projection of the utterance in which the term is uttered onto $\mathbf{w}$ greater than $\theta+1$, and the projection of the utterance in

```
Require: training set $S = \{(\bar{p}^{r_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{y}^{r_i})\}_{i=1}^m$;
  desired threshold $\theta$, trade-off parameter $C$
  $\mathbf{w}_0 = \mathbf{0}$
  for $i = 1, \dots, m$ do
    $\bar{y}'_i = \arg\max_{\bar{y}} \; \mathbf{w}_{i-1}^\top \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{r_i}, \bar{y})$
    $\ell^+ = 1 + \theta - \mathbf{w}_{i-1}^\top \phi(\bar{\mathbf{x}}_i^+, \bar{p}^{r_i}, \bar{y}^{r_i})$
    $\ell^- = 1 - \theta + \mathbf{w}_{i-1}^\top \phi(\bar{\mathbf{x}}_i^-, \bar{p}^{r_i}, \bar{y}'_i)$
    if $\ell^+ > 0$ and $\ell^- > 0$ then
      $\Delta\boldsymbol{\phi} = \phi(\bar{\mathbf{x}}_i^+, \bar{p}^{r_i}, \bar{y}^{r_i}) - \phi(\bar{\mathbf{x}}_i^-, \bar{p}^{r_i}, \bar{y}'_i)$
      $\tau = (\ell^+ \; + \; \ell^-) \, / ||\Delta\boldsymbol{\phi}||^2$
    else if $\ell^+ > 0$ and $\ell^- < 0$ then
      $\Delta\boldsymbol{\phi} = \phi(\bar{\mathbf{x}}_i^+, \bar{p}^{r_i}, \bar{y}^{r_i})$
      $\tau = \ell^+ / ||\Delta\boldsymbol{\phi}||^2$
    else if $\ell^+ < 0$ and $\ell^- > 0$ then
      $\Delta\boldsymbol{\phi} = -\phi(\bar{\mathbf{x}}_i^-, \bar{p}^{r_i}, \bar{y}'_i)$
      $\tau = \ell^- / ||\Delta\boldsymbol{\phi}||^2$
    else
      $\tau = 0$
    end if
    $\mathbf{w}_i = \mathbf{w}_{i-1} + \min\{C, \tau\} \; \Delta\boldsymbol{\phi}$
  end for
  return The average weight vector $\mathbf{w} = \frac{1}{m}\sum_{i=1}^m \mathbf{w}_i$ or
  the best $\mathbf{w}_i$ on a validation set $\mathbf{w} = \arg\max_i Acc_\theta(\mathbf{w}_i)$.
```

Fig. 1. PA-ACC algorithm for maximizing Acc_θ defined in (2).

which the term is not uttered onto $\mathbf{w}$ less than $\theta - 1$. The closed form solution to the above optimization problem can be derived using the Karush-Kuhn-Tucker conditions in the same lines of [16, App. A].

The loss in (10) is composed of two hinge functions and therefore introduces a more elaborate solution than the one derived for the ranking loss of [8]. We call this algorithm PA-ACC (Passive-Aggressive to maximize Accuracy). A pseudo-code of the algorithm is given in Fig. 1. At every iteration i in $[1, ..., m]$, the algorithm calculates the two components of the surrogate loss defined by (10); we denote these two components by ℓ^+ and ℓ^- respectively. The algorithm then assigns values to τ and $\Delta\boldsymbol{\phi}$, which are used to update the weight vector $\mathbf{w}$. These two variables could be viewed as the equivalents of the learning rate and the gradient, respectively. Their values are determined according to the 4 possible cases of whether ℓ^+ or ℓ^- are greater than or less than zero. Finally, the weight vector $\mathbf{w}$ is updated using the values assigned to τ and $\Delta\boldsymbol{\phi}$.

Since the PA-ACC algorithm is an online algorithm which deals with drifting hypotheses, it is highly influenced by the recent examples. Common methods to convert an online algorithm to a batch algorithm are either by taking the average over all the parameters $\{\mathbf{w}_i\}$, or by taking the best $\mathbf{w}_i$ over a validation set [17], [18].

V. Deep network model

We turn to exemplify our idea in the weakly supervised setting using deep networks. This implementation is based on recent work on whole-word segmental systems [11], [12], [13]. These works present a Siamese network model trained with a ranking loss function. Siamese networks [19] are neural networks with a tied set of parameters which take as input a pair of speech segments and are trained to minimize or maximize the similarity between the segments depending on whether the same term has been pronounced in the pair of segments.

In this setting the term r is represented by two speech segments rather than a phoneme sequence: a speech segment in which the term r is pronounced, $\bar{\mathbf{x}}^+$, and a speech segment, in which the term r is not pronounced, $\bar{\mathbf{x}}^-$. Similar to those works, we assume that each example in the training set is composed of the triplet $(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^+, \bar{\mathbf{x}}^-)$, where $\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^+ \in \mathcal{X}^{r+}$ and $\bar{\mathbf{x}}^- \in \mathcal{X}^{r-}$. The goal in training the network is that the similarity score between $\bar{\mathbf{x}}^t$ and $\bar{\mathbf{x}}^+$ should be above the similarity score between $\bar{\mathbf{x}}^t$ and $\bar{\mathbf{x}}^-$.

Denote by $g_\mathbf{u} : \mathcal{X}^* \to \mathbb{R}^d$ a deep network (the specific architecture is discussed in Section VI) with a set of parameters $\mathbf{u}$, where d is the dimension of the output. Denote by $\rho : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a measure of similarity between two output vectors of size d. The spoken term detector $f_\mathbf{u} : \mathcal{X}^* \times \mathcal{X}^* \to \mathbb{R}$ is the composition of Siamese networks $g_\mathbf{u}$ and the similarity function. Hence an unknown speech segment $\bar{\mathbf{x}}^t$ can be compared to a positive speech segment

$$f_\mathbf{u}(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^+) = \rho(g_\mathbf{u}(\bar{\mathbf{x}}^t), g_\mathbf{u}(\bar{\mathbf{x}}^+)),$$

or to a negative speech segment

$$f_\mathbf{u}(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^-) = \rho(g_\mathbf{u}(\bar{\mathbf{x}}^t), g_\mathbf{u}(\bar{\mathbf{x}}^-)).$$

The tied parameters $\mathbf{u}$ of all the models were found in [11], [12], [13] using the minimization of the ranking loss function

$$\ell(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^+, \bar{\mathbf{x}}^-) = \Big[\gamma - f_\mathbf{u}(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^+) + f_\mathbf{u}(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^-)\Big]_+ \qquad (12)$$

for different options of the similarity function ρ. In this work we propose to minimize the loss function in (7), which is defined for the weakly supervised case as follows:

$$\ell(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^+, \bar{\mathbf{x}}^-; \theta) = \Big[\gamma + \theta - f_\mathbf{u}(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^+)\Big]_+ + \Big[\gamma - \theta + f_\mathbf{u}(\bar{\mathbf{x}}^t, \bar{\mathbf{x}}^-)\Big]_+ , \qquad (13)$$

when the margin of $\gamma > 0$ is used. In this case, the parameter γ is not set to 1, since the function f_u is not a linear function and hence is not scale invariant to the margin, as in the structured prediction case.

In the next section we present our empirical comparison for all the loss functions on different speech corpora.

VI. Experiments

In this section we present experimental results that demonstrate the effectiveness of our proposed calibrated loss function (7). We compared the proposed loss to the standard approach of maximizing AUC using the ranking loss as in (12) where no fixed threshold can be set. The experiments on the structured prediction model were conducted using fully supervised training sets of read speech (TIMIT, WSJ). The experiments on the deep network model performed on a weakly supervised data of conversational speech (Switchboard).

A. Structured prediction model

To validate the effectiveness of the proposed approach, we performed experiments with the TIMIT corpus. The training and validation sets were taken from the TIMIT training set. The training set was composed from 1,512 randomly chosen terms, corresponding to 11,139 pairs of positive and negative utterances (each term repeated more than once). Similarly, the validation set was composed from 378 different randomly chosen terms, corresponding to 2,892 pairs. The validation set was used to tune the algorithm's parameters.

The test set was composed of 80 terms that were suggested as a benchmark in [8], and are distinct from the terms used in the training and validation sets. For each term, we randomly picked at most 20 utterances in which the term was uttered and at most 20 utterances in which it was not uttered. The utterances were taken from the TIMIT test set. The number of test utterances in which the term was uttered was not always 20, since some terms were uttered less than 20 times in the whole TIMIT test set.

We measure performance using the AUC defined in (1) and using the accuracy of a fixed threshold θ denoted Acc_θ. Specifically, we calculate AUC on the test set of m_{test} examples according to

$$\widehat{AUC} = \frac{1}{m_{\text{test}}} \sum_{i=1}^{m_{\text{test}}} \mathbb{I}\Big\{ f(\bar{\mathbf{x}}_i^+, \bar{p}^{r_i}) \geq f(\bar{\mathbf{x}}_i^-, \bar{p}^{r_i}) \Big\}, \quad (14)$$

and the accuracy by

$$\widehat{Acc_\theta} = \frac{1}{m_{\text{test}}} \sum_{i=1}^{m_{\text{test}}} \mathbb{I}\Big\{ f(\bar{\mathbf{x}}_i^+, \bar{p}^{r_i}) > \theta \wedge f(\bar{\mathbf{x}}_i^-, \bar{p}^{r_i}) < \theta \Big\}. \quad (15)$$

We tested the PA-ACC algorithm using two options. The first was whether the final weight vector was a result of averaging or was the best to perform on the validation set. The second option was whether the new feature functions we introduced were normalized by the length of the phoneme sequence $|\bar{p}^r|$ or not. The AUC and Acc_θ rates found on our validation and test set are presented in Table I. In training PA-ACC we chose arbitrarily $\theta = 0$.

TABLE I
AUC AND ACC_θ RATES OF THE PA-ACC ALGORITHM. THE FIRST COLUMN INDICATES WHETHER THE NEW FEATURE FUNCTIONS WERE NORMALIZED OR NOT. THE SECOND COLUMN INDICATES WHETHER THE FINAL WEIGHT VECTOR WAS A RESULT OF AVERAGING OR WAS THE BEST TO PERFORM ON THE VALIDATION SET.

Normalized		AUC		Acc_θ	
feature func.	Final $\mathbf{w}$	Validation	Test	Validation	Test
False	best.	0.978	1.000	**0.846**	0.931
False	average.	0.978	1.000	0.829	0.920
True	best.	0.978	1.000	0.813	0.926
True	average.	0.964	0.994	0.811	**0.943**

We can see from the table that since the TIMIT dataset is very clean the detection rates are very good and the AUC is almost always 1. The results presented here are improved over the results presented in [8] due to the introduction of the new feature functions. It is interesting to note that the best Acc_0 results on the validation set were obtained when the additional features were not normalized and the final weight vector was selected over the validation set, while the best Acc_0 results on the test set were obtained with the opposite conditions: when the final weight vector was the average one and the additional feature functions were normalized. Further research on feature functions should be conducted and extended to a larger dataset.

Setting the threshold θ to zero was an arbitrary choice, and changing its value does not affect the Acc_θ and AUC rates. The reason is that the values of the parameters $\mathbf{w}$ can be scaled according to the value of the threshold. Hence the actual value of the threshold does not play an important role. This is similar to the concept of large margin which is arbitrarily set to 1 in the optimization problem and not to a specific value [20].

We now turn to compare the performance of our algorithm against two other algorithms. The first is the discriminative keyword spotting algorithm presented in [8], which is the Passive-Aggressive algorithm trained with the ranking loss to maximize the AUC. It is denoted here as PA-AUC. We introduce two versions of this algorithm: the original version and an extended version with the additional set of feature functions described in Sec. IV. When using the extended version of PA-AUC, normalizing the features had no affect on our results. Similarly, a comparison of using the final weight vector versus the best weight vector yielded similar outcomes.

The second algorithm is an HMM-based spoken term detection algorithm presented in [8][1].

For all the algorithms we report the AUC and Acc_θ in Table II. For the two versions of PA-AUC we selected a single threshold θ that gave the best Acc_θ on the validation set. Similarly we selected the best threshold for the HMM algorithm. For PA-ACC we selected $\theta = 0$.

TABLE II
A COMPARISON OF ACC_θ AND AUC RATES OF 3 ALGORITHMS: PA-AUC, HMM AND PA-ACC ON THE TIMIT CORPUS.

	AUC	Acc_θ
PA-AUC	0.988	0.846
PA-AUC (additional feature func., norm.)	0.994	0.926
PA-AUC (additional feature func., no norm.)	0.994	0.926
HMM	0.943	0.724
PA-ACC (norm, avg)	0.994	**0.943**
PA-ACC (no norm, best)	**1**	0.931

It is interesting to see that the AUC of PA-ACC is the same or even higher than that of the PA-AUC. Since Acc_θ is a lower bound to AUC, the AUC can be thought of as Acc_θ with the best threshold selected for every term in the set. Indeed from Table II we see that the Acc_θ was very close to the AUC but did not reach it.

[1]left-to-right HMM of 5 emitting states with 40 diagonal Gaussians trained as a phoneme recognizer.

We evaluate the model trained on TIMIT on the Wall Street Journal (WSJ) corpus [21]. This corpus corresponds to read articles of the Wall Street Journal, and hence presents a different linguistic context compared to TIMIT. Both the discriminative system and the HMM-based system were trained on the TIMIT corpus as described above and evaluated on a different set of 80 keywords from the WSJ corpus. For each keyword, we randomly picked at most 20 utterances in which the keyword was uttered and at most 20 utterances in which it was not uttered from the `si_tr_s` portion of the WSJ corpus. We used the same setting as in [8]. As before we arbitrarily chose the threshold $\theta = 0$. The results are presented in Table III.

TABLE III
A COMPARISON OF Acc_θ AND AUC RATES OF 3 ALGORITHMS: PA-AUC, HMM AND PA-ACC ON THE WSJ CORPUS.

	AUC	Acc_θ
PA-AUC	0.945	0.803
PA-AUC (additional feature func.)	0.951	0.821
HMM	0.879	0.578
PA-ACC (norm, avg)	0.949	0.827
PA-ACC (no norm, best)	**0.952**	**0.833**

Again we see from Table III that the model trained with the proposed loss function led to higher accuracy rates with similar AUC rates, meaning a better separation between the positive speech utterances and the negative speech utterances.

B. Deep network model

Our second set of experiments is focused on deep networks trained on weakly supervised data. Our model is based on previous work on network training using the ranking loss [11], [12], [13]. We used the same experimental setup as Kamper *et al.* (2016)[13]. By using the term *weak supervision*, we followed the terminology of [13]. Indeed the term *weak* refers to the fact that supervision is given in the form of known word pairs, rather than the exact location of the term and its phonetic content as in Subsection VI-A.

The data was taken from the Switchboard corpus of English conversational telephone speech. Mel-frequency cepstral coefficients (MFCCs) with first and second order derivatives features were extracted and cepstral mean and variance normalization (CMVN) was applied per conversation side. The training set consisted of the set of about 10k word tokens from [22], [23]; it consisted of word segments of at least 5 characters and 0.5 seconds in duration extracted from a forced alignment of the transcriptions, and comprises about 105 minutes of speech. For the Siamese convolutional neural networks (CNNs), this set results in about 100k word segment pairs. For testing, we used the 11k-token set from [22], [23].

The architecture of each network was the same as Kamper *et al.* (2016)[13]: 1-D convolution with 96 filters over 9 frames; ReLU (Rectified Linear Unit) ; max pooling over 3 units; 1-D convolution with 96 filters over 8 units; ReLU; max pooling over 3 units; 2048-unit fully-connected ReLU; 1024-unit fully-connected linear layer. All weights were initialized randomly. Models were trained using ADADELTA [24].

We reproduced the results in [13] by training the Siamese network using the ranking loss in (12) with the cosine similarity as a similarity function ρ. The cosine similarity of two vectors $\mathbf{v}_1 \in \mathbb{R}^d$ and $\mathbf{v}_2 \in \mathbb{R}^d$ is defined as

$$\rho_{\cos}(\mathbf{v}_1, \mathbf{v}_2) = \frac{\mathbf{v}_1^\top \mathbf{v}_2}{\|\mathbf{v}_1\| \|\mathbf{v}_2\|},$$

where this function returns a number close to 1 if the two vectors are similar and a number close to -1 if the two vectors are not. We also train the network using the same similarity function using the Acc_θ loss function with $\theta = 0$ as in (13). For the ranking loss we used $\gamma = 0.15$ while for the Acc_θ loss we used $\gamma = 0.10$. The reason is the margin γ is counted twice for Acc_θ loss. These values were chosen by maximizing Acc_θ on a validation set over 5 epochs. The AUC and Acc_θ values for training of 5 to 30 epochs are given in Table IV. Other training parameters and settings were exactly the same as in [13].

We can see in the table that both AUC and Acc_θ are higher when the system is trained with the calibrated ranking loss function. The reason that the AUC was also improved is most likely because the calibrated ranking loss function is harder to optimize than the ranking loss. Note that for this corpus the score Acc_θ is not close to AUC, meaning there is some room for further improvements. A possible reason for this is that while the hypothesis set is rich enough to describe AUC adequately, it may not be sufficient to express the fixed threshold needed for Acc_θ. Further research could focus on how to direct the network to express more knowledge on the internal representation of the threshold, in a similar fashion to the additional set of feature functions manually added to the structured prediction model.

VII. CONCLUSIONS

In this work, we introduced a new loss function that can be used to train a spoken term detection system with a fixed desired threshold for all terms. We introduced a new discriminative structured prediction model that is based on the Passive-Aggressive algorithm. We show that the new loss can be used in training weakly supervised deep network models. Results suggest that our new loss function yields AUC and accuracy values that are better than previous works' results.

In STD research area, the common approach is to use an LVCSR to transcribe the test corpus to lattices, then apply retrieval techniques on the lattices. In future work we plan to report the results of the LVCSR-based technique using the proposed loss function. Another line of future work will be focused on optimizing the ATWV score (rather than AUC), which is a more common score in STD systems.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their suggestions and comments, which helped us to improve the manuscript. This work was supported by a grant from the MAGNET program of the Israeli Innovation Authority.

TABLE IV
ACC$_\theta$ AND AUC RESULTS FOR ACOUSTIC WORD EMBEDDING ON SWITCHBOARD DATA, USING VARIOUS AMOUNTS OF EPOCHS. THE TRAINING SETS OF ALL EPOCHS ARE SUBSETS OF THE 30 EPOCH TRAINING SET.

	5 Epochs		10 Epochs		15 Epochs		20 Epochs		25 Epochs		30 Epochs	
loss type	AUC	Acc$_\theta$	AUC	Acc$_\theta$	AUC	Acc$_\theta$	AUC	Acc$_\theta$	AUC	Acc$_\theta$	AUC	Acc$_\theta$
ranking loss	0.923	0.703	0.923	0.711	0.925	0.708	0.923	0.707	0.920	0.706	0.925	0.709
Acc$_\theta$ loss	**0.981**	**0.785**	**0.976**	**0.757**	**0.977**	**0.757**	**0.978**	**0.757**	**0.979**	**0.761**	**0.978**	**0.757**

APPENDIX A

In this appendix we provide some theoretical analysis on our proposed method. We start by showing that the accuracy Acc_θ is less than or equal to the AUC for any threshold θ. This means that minimizing $Err_\theta = 1 - Acc_\theta$ can lead to the minimization of $1 - AUC$ or equivalently to maximization of the AUC.

Lemma 1. *For any threshold $\theta \in \mathbb{R}$ and the accuracy Acc_θ is less than or equal the AUC:*

$$Acc_\theta \leq AUC. \tag{16}$$

Proof. Consider the definition of the AUC. In the case in which there exists a fixed but unknown threshold z for all possible terms and the corresponding positive and negative speech segments, the AUC can be written as

$$\mathbb{P}\left\{f(\bar{\mathbf{x}}^+, r) > z \quad \wedge \quad f(\bar{\mathbf{x}}^-, r) < z\right\}.$$

In this case z is the optimal threshold and any other predefined threshold would result in a lower probability hence

$$\begin{aligned}&\mathbb{P}\left\{f(\bar{\mathbf{x}}^+, r) > \theta \quad \wedge \quad f(\bar{\mathbf{x}}^-, r) < \theta\right\}\\ &\qquad \leq \quad \mathbb{P}\left\{f(\bar{\mathbf{x}}^+, r) > z \quad \wedge \quad f(\bar{\mathbf{x}}^-, r) < z\right\}. \end{aligned}\tag{17}$$

In the case when no threshold z exists, the event defined by Acc_θ is still a special case of the event defined by AUC, and hence its probability is smaller than the probability of the AUC. □

We now state two theorems that show the theoretical properties of the algorithm in Fig. 1. Our first theorem shows that the detection function of our algorithm attains high cumulative AUC (defined in (14)) during training. 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 algorithm cannot do much worse than the best fixed weight vector. Formally, it shows that the cumulative area *above* the curve, $1 - \widehat{AUC}_{\text{train}}$, is smaller than the weighted average loss $\ell(\mathbf{w}^\star; (\bar{p}^{r_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{y}^{r_i}))$ 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. *Given a training set of examples $S = \{(\bar{p}^{r_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{y}^{r_i})\}_{i=1}^m$ we assume that for every $\bar{\mathbf{x}}$, $\bar{p}$, and $\bar{y}$ we have $\|\phi(\bar{\mathbf{x}}, \bar{p}, \bar{y})\| \leq 1/2$. Let $\mathbf{w}^\star$ be the best weight vector selected by some optimization criterion by observing all instances in hindsight, we have*

$$1 - \widehat{AUC}_{train} \leq \frac{3}{m}\|\mathbf{w}^\star\|^2 + \frac{2C}{m}\sum_{i=1}^{m}\ell(\mathbf{w}^\star; (\bar{p}^{r_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{y}^{r_i})). \tag{18}$$

where $C \geq 1$ and $\widehat{AUC}_{train}$ is the cumulative AUC defined in (14).

We omit the proof of the theorem, which is based on Lemma 1 and similar to the proof of Theorem 1 in [8].

Our second theorem shows that our structured prediction algorithm generalizes well. That is, the algorithm's AUC values on an unseen test set are likely to be similar to its AUC values during training.

Theorem 2. *Under the same conditions of Theorem 1. Assume that the training set S and the validation set V are both sampled i.i.d. from a distribution D. Denote by m the size of the training set, and by m_v the size of the validation set. With probability of at least $1 - \delta$ we have*

$$\begin{aligned}1 - AUC \leq &\frac{2C}{m}\sum_{i=1}^{m}\ell(\mathbf{w}^\star; (\bar{p}^{r_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{y}^{r_i})) \\ &+ \frac{3}{m}\|\mathbf{w}^\star\|^2 + \frac{\sqrt{2ln(2/\delta)}}{\sqrt{m}} + \frac{\sqrt{2ln(2m/\delta)}}{\sqrt{m_v}},\end{aligned}\tag{19}$$

where the AUC is defined by (1).

The proof of the theorem goes along the same lines as the proof of Theorem 2 in [8].

REFERENCES

[1] S. Bengio, J. Mariéthoz, and M. Keller, "The expected performance curve," in *International Conference on Machine Learning, ICML, Workshop on ROC Analysis in Machine Learning*, no. EPFL-CONF-83266, 2005.

[2] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddingtion, "Results of the 2006 spoken term detection evaluation," in *Proc. SIGIR*, vol. 7, 2007, pp. 51–57.

[3] J. Mamou, B. Ramabhadran, and O. Siohan, "Vocabulary independent spoken term detection," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2007, pp. 615–622.

[4] D. Karakos, R. Schwartz, S. Tsakalidis, L. Zhang, S. Ranjan, T. T. Ng, R. Hsiao, G. Saikumar, I. Bulyko, L. Nguyen *et al.*, "Score normalization and system combination for improved keyword spotting," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 210–215.

[5] C. Parada, A. Sethy, and B. Ramabhadran, “Query-by-example spoken term detection for oov terms,” in *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*. IEEE, 2009, pp. 404–409.

[6] D. R. Miller, M. Kleber, C.-L. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish, “Rapid and accurate spoken term detection,” in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[7] O. Vinyals and S. Wegmann, “Chasing the metric: Smoothing learning algorithms for keyword detection,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3301–3305.

[8] J. Keshet, D. Grangier, and S. Bengio, “Discriminative keyword spotting,” *Speech Communication*, vol. 51, no. 4, pp. 317–329, 2009.

[9] R. Prabhavalkar, J. Keshet, K. Livescu, and E. Fosler-Lussier, “Discriminative spoken term detection with limited data,” in *The 2nd Symposium on Machine Learning for Speech and Language Processing*, 2012.

[10] R. Prabhavalkar, K. Livescu, E. Fosler-Lussier, and J. Keshet, “Discriminative articulatory models for spoken term detection in low-resource conversational settings,” in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 8287–8291.

[11] D. Grangier and S. Bengio, “Learning the inter-frame distance for discriminative template-based keyword detection,” in *International Conference on Speech Processing (INTERSPEECH)*, 2007.

[12] S. Bengio and G. Heigold, “Word embeddings for speech recognition.” in *INTERSPEECH*, 2014, pp. 1053–1057.

[13] H. Kamper, W. Wang, and K. Livescu, “Deep convolutional acoustic word embeddings using word-pair side information,” in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4950–4954.

[14] J. A. Hanley and B. J. McNeil, “The meaning and use of the area under a receiver operating characteristic (roc) curve.” *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[15] D. Bamber, “The area above the ordinal dominance graph and the area below the receiver operating characteristic graph,” *Journal of mathematical psychology*, vol. 12, no. 4, pp. 387–415, 1975.

[16] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, “Online passive aggressive algorithms,” *Journal of Machine Learning Research*, vol. 7, pp. 551–585, 2006.

[17] O. Dekel, J. Keshet, and Y. Singer, “Large margin hierarchical classification,” in *Proceedings of the Twenty-First International Conference on Machine Learning (ICML)*, 2004.

[18] N. Cesa-Bianchi, A. Conconi, and C. Gentile, “On the generalization ability of on-line learning algorithms,” *IEEE Transactions on Information Theory*, vol. 50, no. 9, pp. 2050–2057, September 2004.

[19] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 539–546.

[20] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[21] D. B. Paul and J. M. Baker, “The design for the wall street journal-based csr corpus,” in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[22] A. Jansen, S. Thomas, and H. Hermansky, “Weak top-down constraints for unsupervised acoustic model training.” in *ICASSP*, 2013, pp. 8091–8095.

[23] H. Kamper, M. Elsner, A. Jansen, and S. Goldwater, “Unsupervised neural network based feature extraction using weak top-down constraints,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5818–5822.

[24] M. D. Zeiler, “Adadelta: an adaptive learning rate method,” *arXiv preprint arXiv:1212.5701*, 2012.