ROBUST DISCRIMINATIVE KEYWORD SPOTTING FOR EMOTIONALLY COLORED SPONTANEOUS SPEECH USING BIDIRECTIONAL LSTM NETWORKS

Martin Wöllmer[1], Florian Eyben[1], Joseph Keshet[2], Alex Graves[3], Björn Schuller[1], Gerhard Rigoll[1]

[1]Institute for Human-Machine Communication, Technische Universität München, Germany
[2]Idiap Research Institute, Martigny, Switzerland
[3] Institute for Computer Science VI, Technische Universität München, Germany
woellmer@tum.de

ABSTRACT

In this paper we propose a new technique for robust keyword spotting that uses bidirectional Long Short-Term Memory (BLSTM) recurrent neural nets to incorporate contextual information in speech decoding. Our approach overcomes the drawbacks of generative HMM modeling by applying a discriminative learning procedure that non-linearly maps speech features into an abstract vector space. By incorporating the outputs of a BLSTM network into the speech features, it is able to make use of past and future context for phoneme predictions. The robustness of the approach is evaluated on a keyword spotting task using the HUMAINE Sensitive Artificial Listener (SAL) database, which contains accented, spontaneous, and emotionally colored speech. The test is particularly stringent because the system is not trained on the SAL database, but only on the TIMIT corpus of read speech. We show that our method prevails over a discriminative keyword spotter without BLSTM-enhanced feature functions, which in turn has been proven to outperform HMM-based techniques.

Index Terms— Speech recognition, Robustness, Recurrent neural networks

1. INTRODUCTION

The goal of keyword spotting is to reliably detect the presence of a specific word in a given speech utterance. This is most commonly done with Hidden Markov Models (HMM) [1, 2]. However, the use of HMMs has various drawbacks, such as the need for an adequate "garbage model" to handle non-keyword speech. Designing a garbage-model is a nontrivial problem since the garbage model can potentially model any phoneme sequence — including the keyword itself. Further disadvantages of HMM modeling are the suboptimal convergence of the Expectation Maximization (EM) algorithm to local maxima, the assumption of conditional independence of the observations, and the fact that HMMs do not directly maximize the keyword detection rate.

For these reasons we follow [3] in using a supervised, discriminative approach to keyword spotting, that does not require the use of HMMs. In general, discriminative learing algorithms are likely to outperform generative models such as HMMs since the objective function used during training more closely reflects the actual decision task. The discriminative method described in [3] uses feature functions to non-linearly map the speech utterance, along with the target keyword, into an abstract vector space. It was shown to prevail over HMM modeling. However, in contrast to state-of-the-art HMM recognizers which use triphones to incorporate information from past and future speech frames, the discriminative system does not explicitly consider contextual knowledge. In this work we build in context information by including the outputs of a bidirectional Long Short-Term Memory (BLSTM) recurrent neural network [4, 5] in the feature functions. Similar neural network architectures have been successfully applied to speech or emotion recognition related tasks [6, 5, 7], where they exploit contextual information whenever speech production or perception is influenced by emotion, strong accents, or background noise. In contrast to [6], our keyword spotting approach uses BLSTM for phoneme discrimination and not for the recognition of whole keywords. As well as reducing the complexity of the network, the use of phonemes makes it applicable to any keyword spotting task.

In the experimental section we evaluate the robustness of our discriminative BLSTM keyword spotter on the Belfast Sensitive Artificial Listener (SAL) database [8]. We show that applying BLSTM significantly increases the area under the Receiver Operating Characteristics (ROC) curve, which is a common measure for keyword spotting performance.

The paper is structured as follows: Section 2 describes the training algorithm of our keyword spotter, Section 3 explains the BLSTM architecture, Section 4 introduces the various BLSTM enhanced feature functions, Section 5 presents the experimental setup as well as the keyword spotting results, and conclusions are given in Section 6.

2. DISCRIMINATIVE KEYWORD SPOTTING

The goal of the discriminative keyword spotter applied in this work is to determine the likelihood that a specific keyword is uttered in a given speech sequence. Thereby each keyword k consists of a phoneme sequence $\bar{p}^k = (p_1, ..., p_L)$ with L being the length of the sequence and p_l denoting a phoneme out of the domain $\mathcal{P}$ of possible phoneme symbols. The speech signal is represented by a sequence of feature vectors $\bar{\mathbf{x}} = (\mathbf{x}_1, ..., \mathbf{x}_t, ..., \mathbf{x}_T)$ where T is the length of the utterance. $\mathcal{X}$ and $\mathcal{K}$ mark the domain of all possible feature vectors and the lexicon of keywords respectively. The alignment of the keyword phonemes is defined by the start times s_l of the phonemes as well as by the end time of the last phoneme e_L: $\bar{s}^k = (s_1, ..., s_L, e_L)$. We assume that the start time of phoneme p_{l+1} corresponds to the end time of phoneme p_l, so that $e_l = s_{l+1}$. The keyword spotter f takes as input a feature vector sequence $\bar{\mathbf{x}}$ as well as a keyword phoneme sequence $\bar{p}^k$ and outputs a real valued confidence that the keyword k is uttered in $\bar{\mathbf{x}}$. In order to make the final decision whether k is contained in $\bar{\mathbf{x}}$, the confidence score is compared to a threshold b. The confidence calculation is based on a set of non-linear feature functions $\{\phi_j\}_{j=1}^n$ (see Section 4) which

take a sequence of feature vectors $\bar{\mathbf{x}}$, a keyword phoneme sequence $\bar{p}^k$, and a suggested alignment $\bar{s}_k$ to compute a confidence measure for the candidate keyword alignment.

The keyword spotting algorithm searches for the best alignment $\bar{s}$ producing the highest possible confidence for the phoneme sequence of keyword k in $\bar{\mathbf{x}}$. Merging the feature functions ϕ_j to an n-dimensional vector function $\boldsymbol{\phi}$ and introducing a weight vector $\mathbf{w}$, the keyword spotter is given as

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

Consequently f outputs a weighted sum of feature function scores maximized over all possible keyword alignments. This output then corresponds to the confidence that the keyword k is uttered in the speech feature sequence $\bar{\mathbf{x}}$. Since the number of possible alignments is exponentially large, the maximization is calculated using dynamic programming.

In order to evaluate the performance of a keyword spotter, it is common to compute the Receiver Operating Characteristics curve [1, 2] which shows the true positive rate as a function of the false positive rate. The operating point on this curve can be adjusted by changing the keyword rejection threshold b. If a high true positive rate shall be obtained at a preferably low false positive rate, the area under the ROC curve (AUC) has to be maximized. With $\mathcal{X}_k^+$ denoting a set of utterances that contains the keyword k and $\mathcal{X}_k^-$ a set that does not contain the keyword respectively, the AUC for keyword k is calculated as

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

and can be thought of as the probability that an utterance containing keyword k ($\bar{\mathbf{x}}^+$) produces a higher confidence than a sequence in which k is not uttered ($\bar{\mathbf{x}}^-$). Thereby $\mathbb{I}_{\{\cdot\}}$ denotes the indicator function. When speaking of the average AUC, we refer to

$$A = \frac{1}{\mathcal{K}} \sum_{k \in \mathcal{K}} A_k. \tag{3}$$

In [3] an algorithm for the computation of the weight vector $\mathbf{w}$ in Equation 1 is presented. The algorithm aims at training the weights $\mathbf{w}$ in a way that they maximize the average AUC on unseen data. One training example $\{\bar{p}^{k_i}, \bar{\mathbf{x}}_i^+, \bar{\mathbf{x}}_i^-, \bar{s}_i^{k_i}\}$ consists of an utterance in which keyword k_i is uttered, one sequence in which the keyword is not uttered, the phoneme sequence of the keyword, and the correct alignment of k_i. With

$$\bar{s}' = \arg\max_{\bar{s}} \mathbf{w}_{i-1} \cdot \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s}) \tag{4}$$

representing the most probable alignment of k_i in $\bar{\mathbf{x}}_i^-$ according to the weights $\mathbf{w}_{i-1}$ of the previous training iteration $i-1$, a term

$$\Delta\boldsymbol{\phi}_i = \frac{1}{|\mathcal{X}_{k_i}^+||\mathcal{X}_{k_i}^-|} \Big(\boldsymbol{\phi}(\bar{\mathbf{x}}_i^+, \bar{p}^{k_i}, \bar{s}^{k_i}) - \boldsymbol{\phi}(\bar{\mathbf{x}}_i^-, \bar{p}^{k_i}, \bar{s}') \Big) \tag{5}$$

is computed which is the difference of feature functions for $\bar{\mathbf{x}}_i^+$ and $\bar{\mathbf{x}}_i^-$. For the update rule of $\mathbf{w}$ the Passive-Aggressive algorithm for binary classification (PA-I) outlined in [9] is applied. Consequently $\mathbf{w}$ is updated according to

$$\mathbf{w}_i = \mathbf{w}_{i-1} + \alpha_i \Delta\boldsymbol{\phi}_i \tag{6}$$

whereas α_i can be calculated as

$$\alpha_i = \min \left\{ C, \frac{[1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i]_+}{||\Delta\boldsymbol{\phi}_i||^2} \right\}. \tag{7}$$

The parameter C controls the "aggressiveness" of the update rule and $[1 - \mathbf{w}_{i-1} \cdot \Delta\boldsymbol{\phi}_i]_+$ can be interpreted as the "loss" suffered on iteration i. After every training step the AUC on a validation set is computed whereas the vector $\mathbf{w}$ which achieves the best AUC on the validation set is the final output of the algorithm.

3. BIDIRECTIONAL LSTM

The basic idea of bidirectional recurrent neural networks [10] is to use two recurrent network layers, one that processes the training sequence forwards and one that processes it backwards. Both networks are connected to the same output layer, which therefore has access to complete information about the data points before and after the current point in the sequence. The amount of context information that the network actually uses is learned during training, and does not have to be specified beforehand. This makes bidirectional networks a very flexible tool for sequence labeling, and they have been successfully applied to areas as diverse as protein secondary structure prediction [11] and speech recognition [10].

Analysis of the error flow in conventional recurrent neural nets (RNNs) resulted in the finding that long time lags are inaccessible to existing RNNs since the backpropagated error either blows up or decays over time (vanishing gradient problem). This led to the introduction of Long Short Term Memory (LSTM) RNNs [4]. An LSTM layer is composed of recurrently connected memory blocks, each of which contains one or more recurrently connected memory cells, along with three multiplicative "gate" units: the input, output, and forget gates. The gates perform functions analogous to read, write, and reset operations. More specifically, the cell input is multiplied by the activation of the input gate, the cell output by that of the output gate, and the previous cell values by the forget gate. Their effect is to allow the network to store and retrieve information over long periods of time. If, for example the input gate remains closed, the activation of the cell will not be overwritten by new inputs and can therefore be made available to the net much later in the sequence by opening the output gate. This principle overcomes the vanishing gradient problem and gives access to long range context information.

Combining bidirectional networks with LSTM gives Bidirectional LSTM (BLSTM), which has demonstrated excellent performance in phoneme recognition [5] and keyword spotting [6].

4. FEATURE FUNCTIONS

As mentioned in Section 2, our keyword spotter is based on a set of non-linear feature functions $\{\phi_j\}_{j=1}^n$ that map a speech utterance, together with a candidate alignment, into an abstract vector space. We use $n = 7$ feature functions which proved successful for the keyword spotter described in Section 2 [12]. We experiment with including the output activations of the BLSTM network described in Section 3 into the first feature function. In one variant this is extended to a two-dimensional function, giving in an overall feature dimension of $n = 8$. In what follows we describe five versions of the first feature function, denoted ϕ_{1A} - ϕ_{1E}.

Feature function ϕ_{1A} is the same as used in [3] and is based on the hierarchical phoneme classifier described in [13]. The classifier outputs a confidence $g_p(\mathbf{x})$ that phoneme p is pronounced in $\mathbf{x}$ which

is then summed over the whole phoneme sequence to give

$$\phi_{1A}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=1}^{|\bar{p}|} \sum_{t=s_i}^{s_{i+1}-1} g_{p_i}(\mathbf{x}_t). \quad (8)$$

Unlike ϕ_{1A}, the feature function ϕ_{1B} incorporates contextual information for the computation of the phoneme probabilities by replacing the confidences $g_p(\mathbf{x})$ by the BLSTM output activations $o_p(\mathbf{x})$, thus

$$\phi_{1B}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=1}^{|\bar{p}|} \sum_{t=s_i}^{s_{i+1}-1} o_{p_i}(\mathbf{x}_t). \quad (9)$$

Since the BLSTM outputs tend to produce high-confidence phoneme probability distribution spikes for the recognized phoneme of a frame while all other activations are close to zero, it is beneficial to also include the probability distribution $g(\mathbf{x})$ (which - due to the hierarchical structure of the classifier - consists of multiple rather low-confidence spikes) in the first feature function, as in $\boldsymbol{\phi}_{1C}$ - ϕ_{1E}. Therefore $\boldsymbol{\phi}_{1C}$ expands the first feature function to a two-dimensional function which can be written as

$$\boldsymbol{\phi}_{1C}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \begin{pmatrix} \sum_{i=1}^{|\bar{p}|} \sum_{t=s_i}^{s_{i+1}-1} g_{p_i}(\mathbf{x}_t) \\ \sum_{i=1}^{|\bar{p}|} \sum_{t=s_i}^{s_{i+1}-1} o_{p_i}(\mathbf{x}_t) \end{pmatrix}. \quad (10)$$

Alternatively ϕ_{1D} consists of a linear combination of the distributions $g(\mathbf{x})$ and $o(\mathbf{x})$ so that

$$\phi_{1D}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=1}^{|\bar{p}|} \sum_{t=s_i}^{s_{i+1}-1} G \cdot g_{p_i}(\mathbf{x}_t) + O \cdot o_{p_i}(\mathbf{x}_t), \quad (11)$$

whereas G and O are constant weighting factors.

The function ϕ_{1E} takes the maximum of the distributions $g(\mathbf{x})$ and $o(\mathbf{x})$. This maintains the high-confidence BLSTM output activations as well as the multiple rather low-confidence hypotheses of $g(\mathbf{x})$ for p-t coordinates where $o_{p_i}(\mathbf{x}_t)$ is close to zero:

$$\phi_{1E}(\bar{\mathbf{x}}, \bar{p}, \bar{s}) = \sum_{i=1}^{|\bar{p}|} \sum_{t=s_i}^{s_{i+1}-1} \max\left(g_{p_i}(\mathbf{x}_t), o_{p_i}(\mathbf{x}_t)\right). \quad (12)$$

The remaining feature functions ϕ_2 - ϕ_7 used in this work are the same as in [3]. ϕ_2 - ϕ_5 measure the Euclidean distance between feature vectors at both sides of the suggested phoneme boundaries, assuming that the correct alignment will produce a large sum of distances, since the distances at the phoneme boundaries are likely to be high compared to those within a phoneme. Function ϕ_6 scores the timing sequences based on typical phoneme durations whereas ϕ_7 consideres the speaking rate implied with the candidate phoneme alignment, presuming that the speaking rate changes only slowly over time (see [3] for formulas).

5. EXPERIMENTS AND RESULTS

For the training of our keyword spotter and for the comparison of the different feature functions ϕ_{1A} - ϕ_{1E} we used the TIMIT corpus. The TIMIT training set was divided into five parts whereas 1,500 utterances were used to train the framebased phoneme recognizer of the first feature function. 150 utterances served as training set for the forced alignment algorithm which we applied to initialize the weight vector $\mathbf{w}$ (for details see [12]). 100 sequences formed the validation set of the forced aligner, and from the remaining 1,946 utterances two times [1] 200 samples were selected for training and two times 200 utterances for validation of the keyword spotter. From the TIMIT test set 80 keywords were chosen randomly. For each keyword we selected at most 20 utterances which contain the keyword and 20 which do not contain the keyword. The feature vectors consisted of cepstral mean normalized MFCC features 0 to 12 with first and second order delta coefficients. As aggressiveness parameter C for the update algorithm (see Equation 7) we used $C = 1$. For the training of the BLSTM used for feature functions ϕ_{1B} - ϕ_{1E} we chose the same 1,500 utterances as for the phoneme recognizer of ϕ_{1A}, however we split them into 1,400 sequences for training and 100 for validation. The BLSTM input layer had a size of 39 (one for each MFCC feature) and the size of the output layer was also 39 since we used the reduced set of 39 TIMIT phonemes. Both hidden LSTM layers contained 100 memory blocks of one cell each. To improve generalization, zero mean Gaussian noise with standard deviation 0.6 was added to the inputs during training. We used a learning rate of 10^{-5} and a momentum of 0.9.

In [3] the keyword spotter applying feature function ϕ_{1A} was shown to outperform a state-of-the-art left-right HMM with 5 emitting states and 40 diagonal Gaussians, consisting of two sub HMM models, the keyword model and the garbage model (see Figure 1).

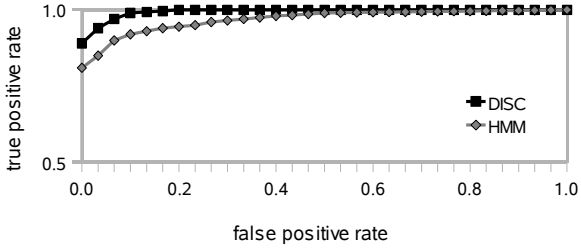

Fig. 1. ROC curve for the discriminative keyword spotter using ϕ_{1A} (DISC) and the HMM approach (results taken from [3])

For feature function ϕ_{1D} we used the parameters G and O that resulted in the best phoneme recognition rate ($G = 1$ and $O = 1.5$). Table 1 shows the average AUC for the different versions of the feature function ϕ_1: best performance is achieved when using ϕ_{1D} or ϕ_{1A}, and there is no statistical significant difference between the result obtained with these two feature functions. Figure 2 illustrates the ROC curve obtained with ϕ_{1D} (DISC-BLSTM) and ϕ_{1A} (DISC) for the TIMIT experiment. Next, we compared the performance of

version of ϕ_1	**AUC**
ϕ_{1D}	0.981
ϕ_{1A}	0.980
ϕ_{1E}	0.970
$\boldsymbol{\phi}_{1C}$	0.965
ϕ_{1B}	0.942

Table 1. AUC for different versions of ϕ_1 (TIMIT experiment)

the keyword spotter using ϕ_{1A} with the best BLSTM keyword spotter using ϕ_{1D} on the Belfast Sensitive Artificial Listener database. In contrast to the TIMIT database which contains read utterances, the SAL corpus contains spontaneous and emotionally colored speech. Note that the SAL utterances have a length of up to 15 seconds which

[1] 200 positive and 200 negative utterances

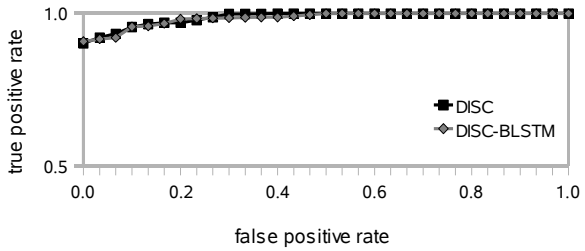

Fig. 2. ROC curve for the discriminative keyword spotter using ϕ_{1A} (DISC) and ϕ_{1D} (DISC-BLSTM)

is longer than the TIMIT sequences, increasing the probability of false positives. For a more detailed description of the SAL database see [8] or [7]. We randomly selected 24 keywords, whereas for each keyword we chose 20 utterances in which the keyword is not uttered and up to 20 utterances (depending on how often the keyword occurs in the whole corpus) which include the keyword. On average, a keyword consisted of 5.4 phonemes. Both the BLSTM network and the keyword spotter were trained on the TIMIT database without any further adaptation to the SAL corpus. For this task our BLSTM approach (using ϕ_{1D}) was able to outperform the keyword spotter which does not use long-range dependencies via BLSTM output activations. The average AUC was 0.80 for the BLSTM experiment and 0.68 for the experiment using the original feature function ϕ_{1A}, respectively. The ROC for both experiments can be seen in Figure 3.

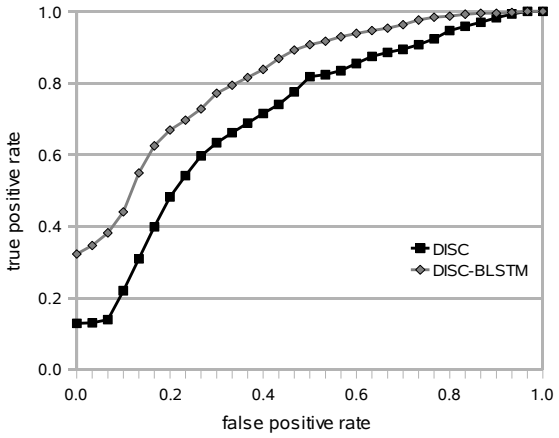

Fig. 3. ROC curve for the SAL experiment applying the BLSTM feature function ϕ_{1D} (DISC-BLSTM) and the original function ϕ_{1A} (DISC)

6. CONCLUSION

This work presented several methods for enhancing the robustness of a discriminative keyword spotter with a BLSTM recurrent neural network. The best method used a modified feature function that included both the phoneme probability scores obtained from a BLSTM network and those given by a hierarchical phoneme classifier. For the TIMIT experiment, both the BLSTM keyword spotter and the non-enhanced version gave almost perfect detection rates. However the BLSTM system gave an 18% improvement in average AUC on the SAL database. This indicates the greater robustness of BLSTM to spontaneous, emotionally colored speech.

For future experiments we will focus on retraining the BLSTM keyword spotter on forced alignments from the SAL database, as a next step towards further improving keyword detection rates for spontaneous emotional speech.

7. ACKNOWLEDGMENT

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 211486 (SEMAINE).

8. REFERENCES

[1] H. Ketabdar, J. Vepa, S. Bengio, and H. Boulard, "Posterior based keyword spotting with a priori thresholds," in *Proceedings of Interspeech*, Pittsburgh, Pennsylvania, 2006.

[2] Y. B. Ayed, D. Fohr, J. P. Haton, and G. Chollet, "Confidence measure for keyword spotting using support vector machines," in *Proceedings of International Conference on Audio, Speech and Signal Processing*, Montreal, Canada, 2004.

[3] J. Keshet, D. Grangier, and S. Bengio, "Discriminative keyword spotting," in *Workshop on Non-Linear Speech Processing NOLISP*, Paris, France, 2007.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9(8), pp. 1735–1780, 1997.

[5] A. Graves, S. Fernandez, and J. Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," in *Proceedings of ICANN*, Warsaw, Poland, 2005, vol. 18, pp. 602–610.

[6] S. Fernandez, A. Graves, and J. Schmidhuber, "An application of recurrent neural networks to discriminative keyword spotting," in *Proceedings of ICANN*, Porto, Portugal, 2007, pp. 220–229.

[7] M. Wöllmer, F. Eyben, S. Reiter, B. Schuller, C. Cox, E. Douglas-Cowie, and R. Cowie, "Abandoning emotion classes - towards continuous emotion recognition with modelling of long-range dependencies," in *Proceedings Interspeech*, Brisbane, Australia, 2008.

[8] E. Douglas-Cowie, R. Cowie, I. Sneddon, C. Cox, O. Lowry, M. McRorie, J. C. Martin, L. Devillers, S. Abrilian, A. Batliner, N. Amir, and K. Karpouzis, *The HUMAINE Database*, vol. 4738, pp. 488–500, 2007.

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

[10] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2673–2681, November 1997.

[11] P. Baldi, S. Brunak, P. Frasconi, G. Soda, and G. Pollastri, "Exploiting the past and the future in protein secondary structure prediction," *BIOINF: Bioinformatics*, vol. 15, 1999.

[12] J. Keshet, *Large Margin Algorithms for Discriminative Continuous Speech Recognition*, Ph.D. thesis, Hebrew University, 2007.

[13] O. Dekel, J. Keshet, and Y. Singer, "Online algorithm for hierarchical phoneme classification," in *Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, Martigny, Switzerland, 2004, pp. 146–159.