Discriminatively learning factorized finite state pronunciation models from dynamic Bayesian networks

Preethi Jyothi[1], *Eric Fosler-Lussier*[1], *Karen Livescu*[2]

[1]Department of Computer Science and Engineering, The Ohio State University, Columbus, Ohio, USA
[2]Toyota Techological Institute at Chicago, Chicago, Illinois, USA

jyothi,fosler@cse.ohio-state.edu, klivescu@ttic.edu

Abstract

This paper describes an approach to efficiently construct, and discriminatively train, a weighted finite state transducer (WFST) representation for an articulatory feature-based model of pronunciation. This model is originally implemented as a dynamic Bayesian network (DBN). The work is motivated by a desire to (1) incorporate such a pronunciation model in WFST-based recognizers, and to (2) learn discriminative models that are more general than the DBNs. The approach is quite general, though here we show how it applies to a specific model. We use the conditional independence assumptions imposed by the DBN to efficiently convert it into a sequence of WFSTs (factor FSTs) which, when composed, yield the same model as the DBN. We then introduce a linear model of the arc weights of the factor FSTs and discriminatively learn its weights using the averaged perceptron algorithm. We demonstrate the approach using a lexical access task in which we recognize a word given its surface realization. Our experimental results using a phonetically transcribed subset of the Switchboard corpus show that the discriminatively learned model performs significantly better than the original DBN.

Index Terms: articulatory features, discriminative training, finite state transducers, dynamic Bayesian networks

1. Introduction

Standard pronunciation models are derived from existing dictionaries with limited means of addressing the various ways in which a word can be pronounced in casual speech [1]. It has been suggested that such phone-based pronunciation models do not properly handle variability due to the coarseness of the phone unit [2, 3]. Modeling speech as multiple streams of linguistically motivated sub-phonetic features has been explored as an alternative, using either hidden Markov models (HMMs) of combined articulatory states [4, 5] or factored-state models represented as dynamic Bayesian networks (DBNs) [6, 7].

The DBN model of pronunciation in [6, 7] is motivated by the theory of articulatory phonology [8]. The locations and constriction degrees of the articulators are represented in the DBN as discrete-valued "articulatory features." In this model, deviations from a canonical pronunciation are the result of either asynchrony between the articulators or substitution of some intended articulatory position with another. The formulation as a DBN (as opposed to an HMM) allows for factorization of a state into multiple variables representing the articulatory features, allowing for a more efficient parameterization (i.e., fewer parameters).

In this work, we develop an approach that keeps the factorization aspect of DBNs, but encodes it in a finite-state transducer representation. The other motivation for our work is a desire to learn the pronunciation model discriminatively, with more flexibility than allowed by generative models like DBNs. The DBN of [6, 7] is trained to maximize the likelihood of the training data. Discriminative approaches typically outperform maximum likelihood parameter estimation in similar settings; the main question is: can we keep the structure of a factored DBN, but allow for discriminative training? We appeal to approaches that implement discriminative training by reweighting scores on the arcs of weighted finite state transducers (WFSTs) [9, 10, 11].

Our approach consists of two components:

- A method for deriving a parsimonious factorized WFST representation (see Fig 1) from a DBN by taking advantage of the conditional independence assumptions encoded in the DBN. This process is detailed in Section 2.1, using the example of a DBN representing an articulatory feature-based pronunciation model (see Fig 2). Using a WFST framework allows us to later take advantage of speech recognition frameworks that represent recognition components (acoustic/pronunciation/language models) as WFSTs.
- An approach to discriminatively learning the weights of the factorized FSTs, described in Section 2.2. This is similar in spirit to previous work on decoding graph optimization techniques that discriminatively learn weights corresponding to the scores in a WFST [12, 9, 10]; for example, Watanabe *et al.* [10] introduce a linear model on the arcs of a decoding graph for a large vocabulary speech recognition task and observe performance improvements comparable to those obtained from discriminative acoustic modeling techniques.

We demonstrate this two-part approach with an isolated-word lexical access task defined in [6]. Our experiments and a discussion of the results are presented in Section 3.

2. Factorized FSTs: Construction and Training

2.1. Efficient conversion of a DBN to factorized FSTs

To illustrate the conversion method, we use a DBN model adapted from [6].[1] Fig 2 is a simplified version of our DBN, corresponding to a single time frame with three articulatory features. This DBN was implemented using the GMTK toolkit [13]. **Word** refers to the word in the current frame. ($\textbf{T-Feat}_1, \textbf{T-Feat}_2, \textbf{T-Feat}_3$) refer to the "target" feature values

[1]The DBN here differs from that of [6] in how it enforces constraints on asynchrony between the articulatory features. In our DBN, the constraints are encoded in the conditional distributions of the $\textbf{Feat}_i\textbf{-Lag}$ variables. This helps to generate smaller WFSTs.

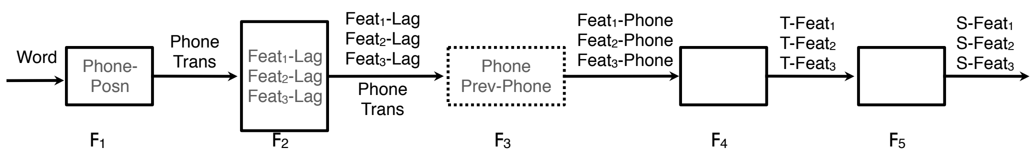

Figure 1: A factorized WFST representation of the DBN in Fig 2 which, when composed from left to right, yields a WFST modeling P($(\textbf{S-Feat}_1, \textbf{S-Feat}_2, \textbf{S-Feat}_3) \mid \textbf{Word}$) of the DBN. Each box denotes a WFST, with input and output variables as shown. The variables shown within the box correspond to those maintained as part of the internal state. The FST shown as a dotted box is deterministic.

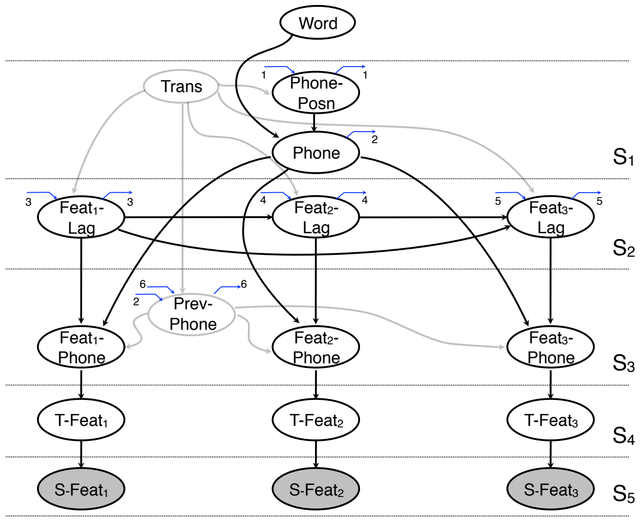

Figure 2: A simplified version of the DBN used in this work (adapted from [6]). We show with dotted lines the partitioning of the random variables into the factors as shown in Fig 1. The numbered edges without parents on a variable indicate a dependency on its value from the previous frame.

for the phones ($\textbf{Feat}_1\textbf{-Phone}, \textbf{Feat}_2\textbf{-Phone}, \textbf{Feat}_3\textbf{-Phone}$) as determined by a phone-to-feature mapping table. ($\textbf{S-Feat}_1, \textbf{S-Feat}_2, \textbf{S-Feat}_3$) refers to the "surface" feature values (actual values produced by the speaker), which are observed for our task (indicated by the shaded variables in Fig 2). In our experiments, we use seven features derived from the vocal tract variables defined in [6, 8] including constriction degrees and locations of the tongue tip, tongue body, lip, glottis and velum. **Phone-Posn** is an index into the underlying phonetic pronunciation of the word (**Phone**) for the current frame ranging from 0 to $N - 1$ where N is the number of phones in the word's pronunciation. However, the actual phones corresponding to each articulatory feature could lag behind **Phone** (and take the value of **Prev-Phone**) thus allowing for asynchrony between the features. These lags for each feature are modeled by ($\textbf{Feat}_1\textbf{-Lag}, \textbf{Feat}_2\textbf{-Lag}, \textbf{Feat}_3\textbf{-Lag}$) (each lag can be either 0 or 1) and the corresponding "feature" phones are ($\textbf{Feat}_1\textbf{-Phone}, \textbf{Feat}_2\textbf{-Phone}, \textbf{Feat}_3\textbf{-Phone}$), respectively. The **Transition** variable helps update the feature lags, **Phone-Posn** and **Prev-Phone** correctly. Henceforth, S will be used to collectively represent all of the surface feature values and W refers to the word. The DBN encodes the joint probability distribution of all random variables; we are mainly interested in the conditional probability $P(S|W)$.

We describe how to find a sequence of WFSTs (using the freely available OpenFst[2]) which when composed (see Fig 2) yields the same model $P(S|W)$ as defined by the DBN. We use this factorized representation rather than a single one because the latter gives rise to an extremely large WFST (call it $\mathcal{AF}$) which is not only computationally infeasible to work with, but also difficult to discriminatively reweight given the limited amount of training data. The algorithm used to derive $\mathcal{AF}$ as a composition of smaller "factor WFSTs" from the DBN does not require the DBN to be explicitly multiplied out and then refactored into factor WFSTs. Rather, the set of random variables in the DBN is partitioned into subsets, $(S_1, \ldots, S_i, \ldots, S_\ell)$, with corresponding factor WFSTs $(F_1, \ldots, F_i, \ldots, F_\ell)$. Each F_i produces variables in S_i that are needed by F_{i+1} as input. All variables needed by F_i to produce its output should be either a part of its input or a part of the internal state representation.

Fig 2 shows the partitioning of its random variables (excluding **Word**) into five subsets:

$$S_1 = \{\textbf{Phone-Posn}, \textbf{Phone}, \textbf{Transition}\}$$
$$S_2 = \{\textbf{Feat}_1\textbf{-Lag}, \textbf{Feat}_2\textbf{-Lag}, \textbf{Feat}_3\textbf{-Lag}\}$$
$$S_3 = \{\textbf{Prev-Phone}, \textbf{Feat}_1\textbf{-Phone}, \textbf{Feat}_2\textbf{-Phone}, \textbf{Feat}_3\textbf{-Phone}\}$$
$$S_4 = \{\textbf{T-Feat}_1, \textbf{T-Feat}_2, \textbf{T-Feat}_3\}$$
$$S_5 = \{\textbf{S-Feat}_1, \textbf{S-Feat}_2, \textbf{S-Feat}_3\}$$

Fig 1 represents the factorized FSTs that we derive using the above partitioning. F_i might have an internal state representation to store the variable values from the previous frame (shown within the box in Fig 1). For example, F_3 uses $\textbf{Feat}_i\textbf{Lag}$ from the current frame, **Prev-Phone** and **Phone** from the previous frame to determine the value of $\textbf{Feat}_i\textbf{Phone}$.

The factor WFSTs are designed as follows:

• $S_1, \ldots, S_5$ are chosen so that $F_1, \ldots, F_5$ are reasonably sized. We require that S_5 has the output random variables (surface feature values in our task) and for each variable in S_i, all its parents should appear in $S_{i-1} \cup S_i$. The sets are judiciously chosen to exploit the fact that, after our discriminative training, the variables within the same set can influence each other beyond what the DBN allowed.

• The inputs and outputs of each F_i are readily determined from the DBN structure, along with what needs to be maintained internally within the states. The arc weights in F_i are determined by the conditional probability tables of the DBN.

• In computing each factor WFST, a dynamic programming (DP) algorithm was used to efficiently determine the weights on the arcs, as products of the appropriate probabilities appearing in the DBN. Each arc weight is a product of several conditional probabilities from the DBN with significant overlap between arcs. Our DP algorithm builds the factor FST incrementally by introducing one random variable at a time such that each conditional probability table from the DBN is read only while introducing that random variable.

These factor FSTs are then discriminatively reweighted by representing their arc weights as a linear model. This is described in more detail in the next section.

[2]http://www.openfst.org/

2.2. Discriminatively training the factorized FSTs

The previous section describes how we produce a set of FSTs $F_1, \ldots, F_\ell$ that represent the probability distribution $P(S|W)$ (derived from the DBN).The decoding process for the lexical access task is to compute the most likely word ($\overline{W}$) for the given surface feature tuple sequence (S); i.e

$$\begin{aligned}
\overline{W} &= \underset{W}{\operatorname{argmax}}\, P(W|S) && (1)\\
&= \underset{W}{\operatorname{argmax}}\, P(S|W) \text{ (assuming uniform prior over words)} && (2)\\
&= \underset{W}{\operatorname{argmax}} \sum_{S_1,\ldots,S_{\ell-1}} P(S_1|W) \prod_{i=1}^{\ell-1} P(S_{i+1}|S_i) P(S|S_1) && (3)\\
&\approx \underset{W}{\operatorname{argmax}} \max_{S_1,\ldots,S_{\ell-1}} P(S_1|W) \prod_{i=1}^{\ell-1} P(S_{i+1}|S_i) P(S|S_1) && (4)\\
&= \text{in}\big[\underset{\vec{a}=\mathbf{a}^1,\ldots,\mathbf{a}^\ell}{\operatorname{argmin}} w(S, \vec{a})\big] && (5)\\
&\quad \text{s.t. out}\big[W \circ \mathbf{a}^1 \circ \ldots \mathbf{a}^i \ldots \circ \mathbf{a}^\ell\big] = S
\end{aligned}$$

where $\vec{a} = \mathbf{a}^1, \ldots, \mathbf{a}^\ell$ is a sequence of paths in the factors $F_1, \ldots, F_\ell$ and out$\big[W \circ \mathbf{a}^1 \circ \ldots \mathbf{a}^i \ldots \circ \mathbf{a}^\ell\big]$ = S iff the input of the arc sequence $\mathbf{a}^1$ is W, the output of $\mathbf{a}^\ell$ is S, and the input of $\mathbf{a}^i$ = output of $\mathbf{a}^{i+1}$. Also, by in[$\vec{a}$] we mean the input sequence of $\mathbf{a}^1$ to get a W sequence. We will soon elaborate on $w(S, \vec{a})$ in Eqn 5. Next, we rewrite $P(S|W)$ in terms of $P(S_i|S_{i-1})$ corresponding to each factor FST. As shown in Eqn 3, we need to sum over all of the different alignments, i.e. settings of the hidden variables such as the feature targets, underlying phones, etc. in the subsets $(S_1, \ldots, S_{\ell-1})$. [3] We make the typical assumption that one alignment is much more likely than all others to derive Eqn 4.

We solve for $\overline{W}$ in Eqn 4 by finding a path tuple ($\vec{a}$) that minimizes the total cost over all paths and satisfies the constraint shown in Eqn 5. $w(S, \vec{a})$ is the total score from F_i for a given S and a tuple of proposed arc sequences, $\mathbf{a}^i$ from each F_i. For the score function, we use a linear model as follows: the score of $\vec{a}$ can be written as the inner product of a feature vector of size $\boldsymbol{N}$ (indexed by $j = 1 \ldots \boldsymbol{N}$), where $\boldsymbol{N}$ is the total number of arcs over all factors, and a corresponding parameter vector $\boldsymbol{\alpha} \in \mathbb{R}^{\boldsymbol{N}}$. The feature vector is scaled by a count function, $\boldsymbol{C}(\vec{a}, r_i)$, that gives the number of times the arc indexed by i (r_i) appears in $\vec{a}$. The feature vector and the score function can be written as follows:

$$\begin{aligned}
\Phi(S, \vec{a}) &:= [\boldsymbol{C}(\vec{a}, r_1)\phi(S, r_1), \ldots, \boldsymbol{C}(\vec{a}, r_{\boldsymbol{N}})\phi(S, r_{\boldsymbol{N}})] && (6)\\
\Rightarrow w(S, \vec{a}) &= \langle \Phi(S, \vec{a}), \boldsymbol{\alpha} \rangle && (7)
\end{aligned}$$

Given N training instances $\{S^n, W^n\}$, $n = 1 \ldots N$, we estimate the weight vector $\boldsymbol{\alpha}$ using Collins' averaged perceptron algorithm [14]. For the n^{th} training instance, Eqn 5 yields the best word hypothesis from the current model, W^n_{dec}, and the best decoded arc sequence tuple, $\vec{a}_{dec} = (\mathbf{a}^1_{dec}, \ldots, \mathbf{a}^\ell_{dec})$. We build a reference finite state acceptor, $\mathcal{W}^n$, that takes as input the correct word label (W^n) for the n^{th} training instance. The arc sequence tuple corresponding to the correct word ($\vec{a}_{corr} = (\mathbf{a}^1_{corr}, \ldots, \mathbf{a}^\ell_{corr})$) is obtained similarly, except we search for these tuples from the composition of the correct word acceptor and the factor FSTs ($\mathcal{W}^n \circ F_1 \circ F_2 \circ \ldots \circ F_\ell$).

If W^n is misrecognized, i.e $W^n \neq W^n_{dec}$, $\boldsymbol{\alpha}$ is updated as:

$$\boldsymbol{\alpha}^{(n+1)} = \boldsymbol{\alpha}^{(n)} + \rho(\boldsymbol{\phi}(S^n, \vec{a}_{dec}) - \boldsymbol{\phi}(S^n, \vec{a}_{corr})) \qquad (8)$$

where ρ is the learning rate. During evaluation, we use the averaged perceptron algorithm that uses $\boldsymbol{\alpha}_{\mathbf{a}^i}$ that is averaged over all training examples and the total number of training epochs

It is instructive to compare our model of a collection of factor FSTs to a single FST $\mathcal{AF}$ that models $P(S|W)$ directly. Our model provides the ability to discriminatively train some factors and not others; this is discussed further in Section 3. Also, using FST factors instead of a single FST could be viewed as a kind of "parameter tying" by which we use a smaller number of weights (on the arcs of the factor FSTs) to derive a large number of weights (on the arcs of the composed FST). One could imagine that the features on each arc a in the composed FST actually consist of a vector of features from the arcs $a^1, \ldots, a^\ell$ that it decomposes into in the factor FSTs. However, we remark that the effect of using the factorized FST model cannot be accurately reproduced by such an approach since the set of arcs $a^1, \ldots, a^\ell$ that a decomposes into is not fixed and depends on the weights themselves.

3. Experiments and Results

3.1. Experimental setup

Our experiments were conducted on a subset of the Switchboard conversational speech corpus that is phonetically labeled at a fine-grained level [15]; these phonetic transcriptions are mapped to articulatory feature value tuples that are used as inputs to our models. The words are excised from continuous utterances and are treated as isolated for our task. The objective of this task is to predict the word given its surface pronunciation. We measure performance by the number of words that are incorrectly predicted (denoted by ER%). There are 3328 words in the dictionary. The parameters of the DBN model were manually initialized based on linguistic judgments (the error rates were not different when we trained the DBN parameters using the Expectation Maximization algorithm).

We use a single feature for arc a^i in factor FST F_i:

$$-\log P(\{\text{out}[a^i], \text{end}[a^i]\} \mid \{\text{in}[a^i], \text{start}[a^i]\})$$

where start$[a^i]$ and end$[a^i]$ refers to the start and end state for arc a^i, respectively and these probabilities are derived from the original DBN model for each arc. By not introducing any additional information to the factorized FST model, other than the DBN probabilities, we ensure that any improvements in performance can be solely attributed to the discriminative nature of the model. We use the same data splits as specified in [6, 7, 16] in our experiments, i.e. a 2942-word training set, a 165-word development set and a 236-word test set.

3.2. Results and discussion

We demonstrate the difficulty of this lexical access task by listing two baseline results (from [17]) on the test set in Table 1. The first baseline model only correctly predicts words in the test set that match baseform pronunciations in the dictionary, giving a large ER of 59.3%. This does not improve considerably when the baseforms are expanded using a large set of phonological rules (ER drops to 56.4%). The articulatory feature-based DBN model shown in Fig 2 reduces the ER considerably. Our

[3]Eqn 3 can be formulated using $P(S_{i+1}|S_i)$ terms because the factor FSTs obey the conditional independence assumptions imposed by the DBN; thus, $P(S_{i+1}|S_i, S_{i-1}, \ldots, S_1) = P(S_{i+1}|S_i)$

Model	ER (%)
Baseform only (from [17])	59.3
Baseform + phonological rules (from [17])	56.4
AF-based DBN (adapted from [6])[a]	43.0
Proposed disc-trained factorized FST model	**35.9**

Table 1: Error rates on the test set.

[a]This number is different from [6] because our model uses a slightly modified DBN structure that makes factorization more intuitive.

discriminatively trained factorized FST models were tuned on the development set and the best results were obtained with $\rho = 0.001$ and 5 training epochs. Our proposed approach reduces ER on the test set by an absolute 7% when compared to the DBN model. This difference is statistically significant at $p < 0.001$ according to McNemar's test.

We can gain insight from examining arcs of a specific factor FST that got reweighted significantly after discriminative training. For example, we analyzed FST F_5 that maps target feature value vectors to surface feature value vectors. One reweighted arc maps an alveolar, uvular sound with a medium degree of opening at both the tongue tip/body (e.g., [ah], [ax]) to a palato-alveolar, uvular sound with a wide and medium-narrow opening of the tongue tip/body, respectively (e.g., [uh]). This transduction becomes far less likely after training, with the weight on this arc being five times larger than the original DBN score. This indicates that a set of surface feature values when grouped together ([uh]) are far less likely given a set of target feature values ([ah]). Our factorized model allows for surface feature values to be correlated given a target feature value configuration, unlike in the DBN.

We chose the lexical access task to demonstrate the feasibility of deriving a discriminatively trained factorized FST, starting with a fairly complex generative (DBN) model. We did not focus on obtaining the best possible model for this task (our ER% on the test set is higher than the reported best results for this task [16]). We intend to incorporate these discriminative finite-state models within a full speech recognition FST cascade with both word-level and acoustically motivated sub-word level features on its arcs.

4. Conclusions

This paper proposes an efficient way of building a factorized finite state model from a DBN that can be discriminatively trained using a linear classifier. We demonstrate our approach using a lexical access task and show significant improvements over the original DBN model. There are a number of ways in which this work could be further extended. Firstly, our model representation allows for it to be easily incorporated within a complete speech recognition system for comparisons with a phone-based recognizer. This representation allows us to include features at both the word and the segment level on the arcs of the factor FSTs. Secondly, the DBN model we use in our experiments allows only context-independent substitutions in each frame, i.e. the surface feature value only depends on the current target feature value. We have observed performance improvements when we use context-dependent substitutions [7], and it will be interesting to observe the benefits of discriminatively training such a model. And lastly, there have been previous attempts at discriminative parameter learning for Bayesian networks [18]. We reserve a comparison of our approach of discriminatively reweighting the arcs of a factorized finite-state model derived from a Bayesian network against these discriminative parameter learning techniques for future work.

5. Acknowledgements

We thank Rohit Prabhavalkar for helpful comments on an earlier draft. This research was supported by the NSF grants IIS-0905633 and IIS-0905420. The opinions expressed in this work are those of the authors and do not necessarily reflect the views of the funding agency.

6. References

[1] D. McAllaster, L. Gillick, F. Scattone, and M. Newman, "Fabricating conversational speech data with acoustic models: A program to examine model-data mismatch," in *Proc. ICSLP*, 1998.

[2] M. Ostendorf, "Moving Beyond the 'Beads-On-A-String' Model of Speech," in *Proc. of ASRU*, 1999.

[3] M. Saraçlar and S. Khudanpur, "Pronunciation change in conversational speech and its implications for automatic speech recognition," *Computer Speech and Language*, vol. 18, no. 4, pp. 375–395, 2004.

[4] L. Deng and D.X. Sun, "A statistical approach to automatic speech recognition using the atomic speech units constructed from overlapping articulatory features," *The Journal of the Acoustical Society of America*, vol. 95, no. 5, pp. 2702–2719, 1994.

[5] M. Richardson, J. Bilmes, and C. Diorio, "Hidden-articulator markov models for speech recognition," *Speech Communication*, vol. 41, no. 2-3, pp. 511–529, 2003.

[6] K. Livescu and J. Glass, "Feature-based pronunciation modeling with trainable asynchrony probabilities," in *Proc. ICSLP*, 2004.

[7] P. Jyothi, K. Livescu, and E. Fosler-Lussier, "Lexical access experiments with context-dependent articulatory feature-based models," in *Proc. ICASSP*, 2011.

[8] C.P. Browman and L. Goldstein, "Articulatory phonology: An overview," *Phonetica*, vol. 49, no. 3-4, pp. 155–180, 1992.

[9] H.K.J. Kuo, B. Kingsbury, and G. Zweig, "Discriminative training of decoding graphs for large vocabulary continuous speech recognition," in *Proc. of ICASSP*, 2007.

[10] S. Watanabe, T. Hori, and A. Nakamura, "Large Vocabulary Continuous Speech Recognition Using WFST-Based Linear Classifier for Structured Data," in *Proc. of Interspeech*, 2010.

[11] M. Lehr and I. Shafran, "Learning a discriminative weighted finite-state transducerfor speech recognition," *IEEE Trans on Audio, Speech, and Language Processing*, vol. 19, no. 5, pp. 1360–1367, 2011.

[12] S.S. Lin and F. Yvon, "Discriminative training of finite state decoding graphs," in *Proc. of Interspeech*, 2005.

[13] "The Graphical Models Toolkit (GMTK)," http://ssli.ee.washington.edu/~bilmes/gmtk/.

[14] M. Collins, "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms," in *Proc. EMNLP*, 2002.

[15] S. Greenberg, J. Hollenback, and D. Ellis, "Insights into spoken language gleaned from phonetic transcription of the Switchboard corpus," in *Proc. ICSLP*, 1996.

[16] H. Tang, J. Keshet, and K. Livescu, "Discriminative pronunciation modeling: A large-margin, feature-rich approach," in *Proc. ACL*, 2012.

[17] K. Livescu, "Feature-based Pronunciation Modeling for Automatic Speech Recognition," *PhD dissertation*, 1999.

[18] J. Su, H. Zhang, C.X. Ling, and S. Matwin, "Discriminative parameter learning for bayesian networks," in *Proc. ICML*, 2008.