Large-Margin Feature Adaptation for Automatic Speech Recognition

Chih-Chieh Cheng #1, Fei Sha *2 and Lawrence K. Saul #3

# *Department of Computer Science and Engineering, University of California, San Diego*
[1] chc028@cs.ucsd.edu
[3] saul@cs.ucsd.edu

* *Department of Computer Science, University of Southern California*
[2] feisha@usc.edu

***Abstract*—We consider how to optimize the acoustic features used by hidden Markov models (HMMs) for automatic speech recognition (ASR). We investigate a mistake-driven algorithm that discriminatively reweights the acoustic features in order to separate the log-likelihoods of correct and incorrect transcriptions by a large margin. The algorithm simultaneously optimizes the HMM parameters in the back end by adapting them to the reweighted features computed by the front end. Using an online approach, we incrementally update feature weights and model parameters after the decoding of each training utterance. To mitigate the strongly biased gradients from individual training utterances, we train several different recognizers in parallel while tying the feature transformations in their front ends. We show that this parameter-tying across different recognizers leads to more stable updates and generally fewer recognition errors.**

I. Introduction

Modern systems for automatic speech recognition (ASR) consist of two interrelated components: a *front end* for signal processing and feature extraction, and a *back end* for statistical inference and pattern recognition. In most systems, the front end computes mel-frequency cepstral coefficients (MFCCs) and higher-order derivatives of MFCCs that capture changes over time [1]. The back end then analyzes and interprets these MFCCs using continuous-density hidden Markov models (CD-HMMs). While the parameters of CD-HMMs are estimated from large amounts of speech, the parameters in the front end are typically fixed a priori and determined by heuristics.

Recent work, however, is blurring the distinction between front and back ends in ASR. In particular, adaptive methods are increasingly being applied at all stages of pattern recognition, from the lowest levels of feature extraction to the highest levels of decision-making. In ASR, these methods include: (i) heteroscedastic linear discriminant analysis (LDA) [2] and neighborhood component analysis [3] to learn informative low dimensional projections of high dimensional acoustic feature vectors; (ii) stochastic gradient and second-order methods to tune parameters related to frequency warping and mel-scale filterbanks ([4], [5]); (iii) maximum likelihood methods for speaker and environment adaptation ([6], [7]) that perform linear transformations of the acoustic feature space at test time; and (iv) extensions of popular frameworks for discriminative training, such as minimum phone error [8] and maximum mutual information [9], to learn accuracy-improving transformations and projections of the acoustic feature space.

Our work in this paper continues this general line of research in end-to-end training of speech recognizers. Specifically, we focus on methods for large-margin training of CD-HMMs. Large-margin methods for ASR have been studied by many researchers in recent years ([10], [11], [12]). Our work considers how to apply large-margin methods to jointly optimize the acoustic features computed by the front end along with the CD-HMM parameters estimated by the back end.

We build on an online algorithm for large-margin training of CD-HMMs proposed earlier this year [13]. In particular, we extend the previous framework to jointly optimize the acoustic features computed by the front end. Essentially, we show how to learn a highly discriminative low dimensional linear projections of MFCCs concatenated from several adjacent analysis windows. The parameters of the linear projection are the elements of a rectangular matrix which must be jointly estimated along with the usual Gaussian mixture parameters for acoustic models. We describe an online algorithm that alternately updates the acoustic-model and projection-matrix parameters. The online algorithm attempts to eliminate the most egregious recognition errors as identified by the worst violations of large-margin constraints.

Optimizing the acoustic features in the front end raises new issues that did not arise in our previous work [13]. First, the optimization landscape becomes considerably more complex. Second, the projection matrix appears to be especially sensitive to the choice of learning rates.

To stabilize the online learning algorithm, we further integrate the idea of parameter-tying. Parameter-tying has been widely used in ASR ([14], [15]) to reduce model footprints and to learn from limited training data. In this work, we train several recognizers in parallel while tying the projection matrix used to compute acoustic features in their front ends. Our experiments show that this form of parameter-tying *across different recognizers* yields consistent improvement beyond the already significant gains of large-margin training.

Our work is distinguished from previous schemes for feature adaptation in three ways. First, we consider how to jointly optimize the parameters in the front end along with the acoustic models in the back end. Second, the feature reweighting is driven by an objective function for large-margin training, which seeks to separate the log-likelihoods of correct and incorrect transcriptions by an amount proportional to their Hamming distance. Third, we explore parameter-tying not across different mixture components or hidden states in the same CD-HMM, but across altogether different recognizers that we train in parallel.

II. Background

In this section, we introduce basic notation and provide necessary background for our work. We begin by reviewing a recently proposed framework for large-margin training of CD-HMMs. We then review previous approaches for combining, adapting, and optimizing the acoustic features computed by standard front ends for ASR.

A. Large margin HMMs

In ASR, we seek to model joint distributions $\mathcal{P}(\mathbf{s},\mathbf{x})$ over sequences of hidden (phonetic) states $\mathbf{s}=\{s_1,s_2,\cdots,s_T\}$ and acoustic observations or feature vectors $\mathbf{y}=\{y_1,y_2,\cdots,y_T\}$. In HMMs, the joint distribution is parameterized by an initial state distribution $\mathcal{P}(s_1)$, state transition probabilities $\mathcal{P}(s_{t+1}|s_t)$, and emission densities $\mathcal{P}(y_t|s_t)$. Concretely,

$$\mathcal{P}(\mathbf{y},\mathbf{s})=\mathcal{P}(s_1)\prod_{t=1}^{T-1}\mathcal{P}(s_{t+1}|s_t)\prod_{t=1}^{T}\mathcal{P}(y_t|s_t). \qquad (1)$$

In CD-HMMs, the emission densities are typically parameterized by Gaussian mixture models (GMMs):

$$\mathcal{P}(y|s) \;=\; \sum\nolimits_c \mathcal{P}(c|s)\mathcal{P}(y|s,c) \qquad (2)$$

where c indexes the mixture components. The individual component distributions are given by:

$$\mathcal{P}(y|s,c) \;=\; (2\pi)^{-d/2}|\Sigma_{sc}|^{-1/2}e^{-\frac{1}{2}(y-\mu_{sc})^\top\Sigma_{sc}^{-1}(y-\mu_{sc})}. \qquad (3)$$

The problem of learning in CD-HMM is to estimate the Gaussian means μ_{sc}, covariance matrices Σ_{sc}, and mixture weights $\mathcal{P}(c|s)$, as well as the transition probabilities $\mathcal{P}(s_{t+1}|s_t)$. In this paper, we focus on learning the GMM parameters which play the dominant role in acoustic modeling.

Modern ASR increasingly relies on error-driven methods for discriminative training of acoustic models ([16], [17], [18], [19], [11]). Here we briefly review a particular reparameterization of CD-HMMs that has proven useful in recent studies of discriminative training ([10], [20]). Let

$$\gamma_{sc} \;=\; \log\mathcal{P}(c|s)-\log[(2\pi)^{d/2}|\Sigma_{sc}|^{1/2}] \qquad (4)$$

denote the log of the scalar prefactor that normalizes each Gaussian distribution in eq. (2). For each Gaussian mixture component, consider the matrix:

$$\Phi_{sc} \;=\; \left[\begin{array}{cc}\Sigma_{sc}^{-1} & -\Sigma_{sc}^{-1}\mu_{sc}\\ -\mu_{sc}^\top\Sigma_{sc}^{-1} & \mu_{sc}^\top\Sigma_{sc}^{-1}\mu_{sc}+\gamma_{sc}\end{array}\right]. \qquad (5)$$

Note that in terms of this matrix, we can write the Gaussian distribution in state s as:

$$\mathcal{P}(y|s)=\sum_c e^{-\frac{1}{2}z^\top\Phi_{sc}z} \quad \text{where} \quad z=\begin{bmatrix}y\\1\end{bmatrix}. \qquad (6)$$

In our work, we will directly optimize the matrices Φ_{sc}, as opposed to the original Gaussian means μ_{sc} and covariance matrices Σ_{sc}.

The parameterization in eq. (6) was developed for large-margin training ([10], [13]) of CD-HMMs. In large-margin training for ASR, we seek not only to minimize the empirical error rate, but also to separate the scores of correct and incorrect transcriptions by a large amount. To formalize this notion, we define the discriminant function:

$$\mathcal{D}(\mathbf{y},\mathbf{s})=\log\mathcal{P}(s_1)+\sum_{t=1}^{T-1}\log\mathcal{P}(s_{t+1}|s_t)+\sum_{t=1}^{T}\log\mathcal{P}(y_t|s_t). \qquad (7)$$

The discriminant function computes the logarithm of the joint probability in eq. (1) for a particular sequence of acoustic feature vectors $\mathbf{y}$ and hidden states $\mathbf{s}$. Let $\mathbf{r}$ denote the ground truth transcription of the utterance with these acoustic feature vectors. For correct recognition *by a large margin*, we seek parameters for which:

$$\forall\mathbf{s}\neq\mathbf{r}, \quad \mathcal{D}(\mathbf{y},\mathbf{r})>\mathcal{D}(\mathbf{y},\mathbf{s})+\rho\mathcal{H}(\mathbf{s},\mathbf{r}), \qquad (8)$$

where $\mathcal{H}(\mathbf{s},\mathbf{r})$ is the Hamming distance between two hidden state sequences and $\rho>0$ is a constant margin scaling factor. In other words, for large-margin training, the score of the correct transcription should exceed the score of any incorrect transcription by an amount that grows in proportion to the number of recognition errors. Proportional margin-based constraints of this form have been shown to yield improvements even beyond other popular forms of discriminative training ([12], [10]).

B. Feature reweighting

In most modern systems for ASR, the front end computes acoustic feature vectors from mel-frequency cepstral coefficients (MFCCs). Typically, the first $d_0=13$ MFCCs are used in this analysis. Due to co-articulation and other temporal effects, the MFCCs computed in one analysis window may contain information about the phonetic content in neighboring windows. To capture this information, most front ends also incorporate MFCCs from neighboring windows into their acoustic feature vectors. In particular, they compute derivative features, such as delta and delta-delta MFCCs, and augment the feature vector to include them.

The derivative features are computed by linearly combining MFCCs from neighboring analysis windows. The weights used to combine adjacent MFCCs are fixed and determined heuristically. Unlike most other parameters in modern speech recognizers, these weights in the front end are not typically adapted to optimize performance. However, feature adaptation has been studied more generally to improve different stages of ASR, from acoustic modeling to decoding-based transcription. Related work in this area has investigated how to optimize and/or adapt acoustic features when they are used in conjunction with CD-HMMs in the back end ([6], [7]).

In this paper, we consider how to optimize the weights used to compute derivative features in conjunction with the back end for large-margin CD-HMMs (reviewed in section II-A). The standard derivative features are computed from a linear transformation of the raw MFCCs in nearby frames. Let u_t denote the $d_0 = 13$ MFCCs computed at time t, and let v_t denote the "stacked" MFCCs obtained by concatenating $4K+1$ consecutive frames $u_{t-2K}, u_{t-2K+1}, ..., u_t, ..., u_{t+2K}$ for some small value of K. (Here we follow the standard convention that if K frames are used on either side of t to estimate the first-order derivatives, then $2K$ frames are used on either side of t to estimate the second-order derivatives.) Finally, let y_t denote the acoustic feature vector derived from the MFCCs at time t and their first and second-order derivatives. Then y_t and v_t are related by the linear transformation:

$$y_t = H_0 v_t, \tag{9}$$

where H_0 is the projection matrix whose entries approximate derivatives by finite differencing operations on nearby frames.

The matrix H_0 is only one of many possible projection matrices that can be used to compute acoustic feature vectors from MFCCs in adjacent frames of speech. In this paper, we explore different feature reweighting strategies for ASR. In particular, we consider how to learn more general projection matrices in the context of large-margin training for CD-HMMs.

III. Model and training

In this section, we extend our previous framework [13] for large-margin training of CD-HMMs to incorporate the reweighting of acoustic features in the front end. We begin by expressing a cost function for learning in terms of the CD-HMM parameter matrices Φ_{sc} from section II-A and the feature projection matrix H_0 from section II-B. We can use an online algorithm to update the elements of these matrices after the decoding of each training utterance. In practice, however, we find that the projection matrix is very sensitive to the fluctuations that arise in online training. To mitigate the strongly biased gradients from individual training utterances, we train several different recognizers in parallel while tying the feature projection matrices in their front ends. The goal of parameter-tying is to stabilize the optimization by accumulating gradients across different recognizers.

A. Large margin cost function

Our approach builds on the large-margin CD-HMMs described in section II-A. Let x denote a stacked feature vector of MFCCs from $4K + 1$ adjacent windows, as described in section II-B, and let z denote the lower dimensional acoustic feature vector that appears in eq. (5). We seek a projection matrix $H \in \Re^{D \times d}$ that maps the high-dimensional vector x of stacked MFCCs to the low-dimensional acoustic feature vector z; then for each window, we can compute:

$$z = Hx, \quad \text{where} \quad x = \begin{bmatrix} v \\ 1 \end{bmatrix}. \tag{10}$$

Note that H has one extra row and column than the projection matrix H_0 in eq. (9) due to the augmented feature vector z that appears in eq. (5) for large-margin CD-HMMs. In particular, we have $d = 3d_0 + 1$ and $D = (4K+1)d_0 + 1$, where $d_0 = 13$ is the number of MFCCs computed per window.

For large-margin training, we adapt the projection matrix H and the parameter matrices Φ_{sc} so that the constraints in eq. (8) are satisfied for as many training utterances as possible. Let $\{(\mathbf{x}_n, \mathbf{r}_n)\}_{n=1}^N$ denote the N labeled feature-state sequences in the training corpus. For online learning, we examine one utterance at a time and compute the hidden state sequence:

$$\mathbf{s}_n^* = \text{argmax}_{\mathbf{s}} \left[\mathcal{D}(\mathbf{x}_n, \mathbf{s}) + \rho \mathcal{H}(\mathbf{s}, \mathbf{r}_n)\right], \tag{11}$$

where $\mathcal{H}(\mathbf{s}, \mathbf{r})$ is the Hamming distance between two hidden state sequences and $\rho > 0$ is the margin scaling factor. (In principle, the value of ρ should be tuned on held-out utterances; for the experiments in this paper, however, we simply set $\rho = 1$, which we knew to have performed well in previous, related work [13].) Eq. (11) returns either the target state sequence $\mathbf{s}_n^* = \mathbf{r}_n$ if Viterbi decoding yields the correct transcription; otherwise, it returns the hidden state sequence that most egregiously violates the large-margin constraint. Note that $\mathbf{s}_n^*$ can be computed by a simple variant of the dynamic programming procedure for Viterbi decoding. The computation is tractable because the Hamming distance can be written as a sum of costs at individual time steps.

In general, it is not possible for a model to satisfy all the large-margin constraints in eq. (8). We use the following loss function [10] to measure the total constraint violation across the entire training corpus:

$$\mathcal{L}(H, \Phi) \; = \; \sum_n \left[\mathcal{D}(\mathbf{x}_n, \mathbf{s}_n^*) - \mathcal{D}(\mathbf{x}_n, \mathbf{r}_n)\right]^+, \tag{12}$$

where $[z]^+ = \max(z, 0)$ denotes the hinge function. The right hand side of eq. (12) computes a weighted count of the training utterances that do not satisfy the margin constraints in eq. (8). In particular, each utterance is weighted by the margin violation of its worst offending state sequence, as determined by eq. (11).

The margin-based loss function in eq. (12) depends on the matrices Φ_{sc} and H through eqs. (6-7) and (10). Specifically, we can write the discriminant function as:

$$\begin{aligned} \mathcal{D}(\mathbf{y}, \mathbf{s}) \quad &= \quad \log \mathcal{P}(s_1) + \sum_{t=1}^{T-1} \log \mathcal{P}(s_{t+1}|s_t) \\ &\quad + \sum_{t=1}^{T} \log \sum_c e^{-\frac{1}{2} x_t^\top H^\top \Phi_{sc} H x_t}. \end{aligned} \tag{13}$$

Note that while eq. (13) depends on the high dimensional (stacked) cepstral feature vectors $x_t \in \Re^D$, the computation can be performed entirely in terms of the low dimensional features $z_t = Hx_t$. In fact, we can view $H^\top \Phi_{sc} H$ as storing a low-rank factorization of an inverse covariance matrix in the high dimensional space of unprojected cepstral features.

B. Parameter-tying

The loss function in eq. (12) can be minimized by alternately updating H and Φ_{sc}. To minimize eq. (12) in this way, we will extend a recently proposed online algorithm [13] for large-margin training of CD-HMMs. We will give more details of the online algorithm in the next section. However, in this section, we introduce a form of parameter-tying that helps to mitigate the strongly biased gradients from individual training utterances.

We have noticed that small changes in the projection matrix H can drastically change the decoding results. This sensitivity is to be expected since the projection matrix H is used to calculate acoustic features in every frame of speech. One way to reduce this sensitivity is to perform some sort of averaging. Batch training reduces this sensitivity by averaging over multiple training utterances. However, batch training does not scale well to very large data sets, nor does it exploit the fact that many training utterances convey redundant information. For online training, we need a different option. Thus we investigate tying the projection matrix H across several different recognizers whose parameters are jointly updated after decoding each training utterance. By averaging the gradients across multiple recognizers, we hope to obtain more stable online updates.

Parameter-tying in CD-HMMs has been widely adopted for ASR ([14], [15]). Our scheme for parameter-tying is subtly different than previous approaches. Typically, parameters are tied across different hidden states or mixture components in the same recognizer. In our scheme, however, we tie parameters across multiple different recognizers that are trained in parallel. These recognizers may have different model sizes (i.e., different numbers of hidden states and/or mixture components). By tying the projection matrix, however, we force all the recognizers to use the same front end.

Our approach is based on a global cost function for parallel training of multiple models or recognizers. Indexing each available model by $\mathcal{M}$, we write the global cost function as:

$$\mathcal{L}(H,\Phi) \;=\; \sum_{\mathcal{M}}\sum_{n}\left[\mathcal{D}_{\mathcal{M}}(\mathbf{x}_n,\mathbf{s}_n^*)-\mathcal{D}_{\mathcal{M}}(\mathbf{x}_n,\mathbf{r}_n)\right]^{+}, \quad (14)$$

In our implementation, the available models are large-margin CD-HMMs with one hidden state per phone but different numbers of Gaussian mixture components per hidden state. Eq. (14) differs from eq. (12) only in the accumulation of information across models. In fact, the parameter-tying only affects the gradients for optimizing the projection matrix H, but not the gradients for optimizing the individual (non-tied) parameter matrices of each model.

C. Online algorithm

The objective function in eq. (14) lends itself to an alternating minimization procedure. Such a procedure alternates between two phases, one optimizing Φ while holding H fixed; the other optimizing H while holding Φ fixed. We gain some insight into our problem by considering the special case where each hidden state only has one Gaussian mixture component. In this case, for fixed H, the objective function over Φ is piecewise linear and convex; however, for fixed Φ, the objective function over H is piecewise (indefinite) quadratic and no longer convex. Thus the optimization is susceptible to spurious local minima, and we must consider carefully how to initialize the projection and parameter matrices in this context.

We explore how to minimize the tied loss function in eq. (14) using an online learning algorithm. The algorithm is inspired by earlier work on perceptron-style updates for both discrete [21] and continuous-density HMMs [13].

The online algorithm for alternating minimizations works as follows. We choose an utterance $(\mathbf{x}_n,\mathbf{r}_n)$ at random from the training corpus. Then, for each individual model $\mathcal{M}$, we update its parameter matrix $\Phi_{\mathcal{M}}$ by:

$$\Phi_{\mathcal{M}} \;\leftarrow\; \Phi_{\mathcal{M}} \;+\; \eta_{\Phi}\,\frac{\partial}{\partial \Phi_{\mathcal{M}}}\left[\mathcal{D}_{\mathcal{M}}(\mathbf{x}_n,\mathbf{r}_n)-\mathcal{D}_{\mathcal{M}}(\mathbf{x}_n,\mathbf{s}_n^*)\right], \quad (15)$$

where the state sequence $\mathbf{s}_n^*$ is computed from the margin-based Viterbi decoding in eq. (11). The right hand side of eq. (15) depends on the current value of the parameter matrix $\Phi_{\mathcal{M}}$ and the projection matrix H; note that different models are not coupled by this update. Following this update, we choose another utterance $(\mathbf{x}_{n'},\mathbf{r}_{n'})$ at random from the training corpus. We then update the projection matrix H by:

$$H \;\leftarrow\; H \;+\; \eta_{H}\,\frac{\partial}{\partial H}\sum_{\mathcal{M}}\left[\mathcal{D}_{\mathcal{M}}(\mathbf{x}_{n'},\mathbf{r}_{n'})-\mathcal{D}_{\mathcal{M}}(\mathbf{x}_{n'},\mathbf{s}_{n'}^*)\right]. \quad (16)$$

The right hand side of eq. (16) depends on the current value of the projection matrix H and the parameter matrices Φ. Note that unlike the update in eq. (15), all models contribute to the optimization of the projection matrix H through the summation in the gradient. We repeat these updates with many training utterances, alternately updating the GMM and projection matrix parameters. The scalar learning rates η_Φ and η_H determine the step sizes; in practice, we tune them independently to achieve the fastest convergence.

IV. Experiments

We experimented on the TIMIT speech corpus [22] with the algorithms described in the previous section. We first describe the basic framework used to evaluate these algorithms, then present and interpret our experimental results.

A. Evaluation

We used the same methodology as previous benchmarks on the TIMIT speech corpus [10]. We followed the standard partition of the TIMIT corpus for training, testing and validation. The data in the TIMIT corpus is manually segmented and aligned with phonetic transcriptions. We built recognizers using monophone CD-HMMs in which each of 48 states represented a context-independent phoneme. We experimented with models of different sizes by varying the number of Gaussian mixture components in each state. For the phone grammar, we used a simple maximum-likelihood bigram model; in addition, we used the validation data to tune a grammar weight that multiplied the log-transition probabilities for Viterbi and margin-based decoding. We evaluated the performance of each CD-HMM by comparing the hidden state sequences inferred by Viterbi decoding to the "ground-truth" phonetic transcriptions provided by the TIMIT corpus. We report two types of errors: the frame error rate (FER), computed simply as the percentage of misclassified frames, and the phone error rate (PER), computed from the edit distances between ground truth and Viterbi decodings. In calculating the errors, we followed the standard of mapping 48 phonetic classes down to 39 broader categories [23].

Our experiments had two main goals: first, to test whether feature reweighting can improve phoneme recognition beyond the usual gains of discriminative training; second, to investigate the potential benefits of parameter-tying in this context. Our baseline systems were discriminatively trained CD-HMMs with traditional cepstra, delta-cepstra, and delta-delta-cepstra as features [13]. Our front end computed $d_0 = 13$ mel-frequency cepstral coefficients (MFCCs) in each analysis window; initial acoustic features were computed by linearly combining the cepstra across 13 consecutive analysis windows (i.e., including six windows on each side of the current window); see eq. (9). To learn reweighted acoustic features, we concatenate all 169 cepstral features from these 13 windows, append a constant scalar feature of value one, and then estimate a 40x170 projection matrix, as in eq. (10). We experimented on CD-HMMs of different sizes, with 1, 2, 4, or 8 Gaussian mixture components per hidden state.

We report results comparing several different models and training procedures. First, we report the performance of baseline CD-HMMs trained by maximum likelihood estimation. Next, we report the performance of discriminatively trained CD-HMMs without feature reweighting, using an online algorithm for large-margin training [13]. Finally, we report the performance using the alternating online updates in eqs. (15–16), both with and without parameter-tying of the projection matrix H across different models.

Since the optimization for acoustic feature reweighting is non-convex, the results can be sensitive to how model parameters are initialized and updated. We used the following scheme to obtain the positive results in this paper. First, we initialized all discriminatively trained models by their maximum likelihood counterparts. Second, we initialized all models with feature reweighting by setting the upper left block of H equal to H_0; thus, the MFCCs from different windows were initially combined by computing standard *delta* and *delta-delta* features. Third, in some experiments, we constrained the initially zero elements of the projection matrix H to remain zero; in other words, though the features were reweighted, the sparsity pattern of the projection matrix was not allowed to change during learning. This constraint led to more reliable convergence in the models without parameter-tying.

B. Effects of Feature Reweighting and Parameter-Tying

Table I compares the frame and phone error rates of CD-HMMs trained in different ways: by maximum likelihood (ML) estimation, by large-margin (LM) training [13], by large-margin training with feature reweighting (LM+FR) but no parameter-tying, and by large-margin training with feature reweighting and parameter-tying across models of different sizes (LM+FR+PT), using both sparse and full projection matrices H. All discriminatively trained CD-HMMs were initialized by ML estimation to ensure that they started from the same baseline with exactly the same performance.

The results in Table I show three general trends: first, that feature reweighting (LM+FR) improves performance beyond the already significant gains from large-margin training (LM); second, that feature reweighting works best in conjunction with parameter-tying (LM+FR+PT) across different models; third, that the most general scheme for feature reweighting (without sparsity constraints on H) leads to the most improvement, provided that the learning is regularized in other ways. In particular, to obtain the results in the last column of Table I, we not only tied the full matrix H across different models; we also employed a parameter-averaging update for the full matrix H, as described in eq. (7) of earlier work [13]. Without both parameter-tying across models and parameter-averaging over time, learning with full matrices H yielded worse results on both the development and test sets.

The exceptions to these trends are also revealing. For example, in the largest model with 8 Gaussian mixture components per hidden state, the frame and phone error rates are not improved by feature reweighting without parameter-tying; in fact, they are slightly worse. The worse performance may be due to overfitting and/or unreliable convergence. However, the performance in this model is improved when the feature reweighting in the front end is tied across different recognizers. The parameter-tying appears to mitigate the challenges of feature reweighting in large models. Specifically, it appears to dampen the fluctuations that arise in online learning, when updates are based on the decoding of individual training utterances. By tying the projection matrix across different model sizes, the larger model benefits from information that is accumulated across different recognizers.

Finally, we comment on convergence issues. In general, parameter-tying led to better results but not necessarily faster training: that is, roughly the same number of passes through the training data were required to converge (as measured by performance on the validation set). However, while the update rule in eq. (16) accumulates information across different models, we can always distribute the computation across multiple nodes, summing up the gradients from individual models as necessary. When implemented in this way, the total running time for learning is essentially equal to the individual running time of the largest model in the ensemble of recognizers.

V. Discussion

In this paper we have explored how to optimize the acoustic features computed by front ends for ASR. Extending a previously proposed framework for large-margin training of CD-HMMs, we showed that standard acoustic features could be discriminatively reweighted to improve performance. Our best results were obtained by tying the feature reweighting parameters across multiple recognizers and training these different recognizers in an integrated manner. The parameter-tying

TABLE I

Frame and phone error rates on the TIMIT test set for CD-HMMs of varying size, as obtained by maximum likelihood (ML) estimation, large-margin training (LM), feature reweighting (FR), and parameter-tying (PT). See text for details. The best results in each row are shown in bold.

#	Frame Error Rate (%)				
of	H_0		sparse H		full H
mix	ML	LM	LM+FR	LM+FR+PT	LM+FR+PT
1	39.7	30.5	30.4	**29.2**	**29.2**
2	36.2	29.4	28.1	28.1	**27.8**
4	33.1	28.3	**27.4**	**27.4**	27.5
8	30.7	27.3	27.4	26.6	**26.4**

#	Phone Error Rate (%)				
of	H_0		sparse H		full H
mix	ML	LM	LM+FR	LM+FR+PT	LM+FR+PT
1	41.5	32.8	32.2	31.9	**31.5**
2	38.0	31.4	29.6	30.3	**29.5**
4	34.9	30.3	29.3	29.2	**29.1**
8	32.3	28.6	28.8	27.8	**27.7**

across models was used to average the strongly biased gradients from individual training utterances in online learning.

There are many interesting directions for future work. For example, in this paper, when we trained several different recognizers in parallel, we weighted their loss functions equally; in many applications, however, we may care more about the performance of one recognizer than another. Suppose in particular that we are deploying an application to a host with limited resources. In this case, it may be more important to optimize the performance of smaller models than larger ones. More work is needed in this direction.

Another interesting direction for future work is to explore different and more general sets of acoustic features. For example, the cepstra themselves are computed from a linear transformation of the log-magnitude spectra. We could use these log-magnitude spectra as the high dimensional features instead of the cepstra and still initialize our models with the baseline performance of existing approaches. The projection matrices in this case would not only be significantly larger, but more or less completely dense. Presumably, larger training corpora would be required to estimate discriminative projection matrices in this case. However, this regime is precisely where one expects the biggest pay-off from online methods, as we have considered in this paper.

Acknowledgment

Fei Sha would like to acknowledge useful discussions with Samy Bengio, Li Deng and Brian Kingsbury. This work was supported in part by NSF Award 0812576. Fei Sha is also partially supported by the Charles Lee Powell Foundation.

References

[1] L. Rabiner and B.-H. Juang, *Fundamentals of speech recognition*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[2] G. Saon, M. Padmanabhan, R. Gopinath, and S. Chen, "Maximum likelihood discriminant feature spaces," in *Proceedings of the International Conference of Acoustic, Speech and Signal Processing (ICASSP-00)*, vol. 2, 2000, pp. 1129–1132.

[3] N. Singh-Miller, M. Collins, and T. J. Hazen, "Dimensionality reduction for speech recognition using neighborhood components analysis," in *Proceedings of Interspeech-07*, 2007, pp. 1158–1161.

[4] K. Visweswariah and R. Gopinath, "Adaptation of front end parameters in a speech recognizer," in *Proceedings of the International Conference on Spoken Language Processing*, 2004, pp. 21–24.

[5] S. Balakrishnan, K. Visweswariah, and V. Goe, "Stochastic gradient adaptation of front-end parameters," in *Proceedings of Interspeech-2004*, 2004, pp. 1–4.

[6] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, 1998.

[7] A. Stolcke, L. Ferrer, S. Kajarekar, E. Shriberg, and A. Venkataraman, "MLLR transforms as features in speaker recognition," in *Proceedings of INTERSPEECH-2005*, 2005, pp. 2425–2428.

[8] B. Zhang and S. Matsoukas, "Minimum phoneme error based heteroscedastic linear discriminant analysis for speech recognition," in *Proceedings of the International Conference of Acoustic, Speech and Signal Processing (ICASSP-05)*, 2005, pp. 925–928.

[9] J. McDonough, M. Wlfelc, and E. Stoimenov, "On maximum mutual information speaker-adapted training," *Computer Speech and Language*, vol. 22, no. 2, pp. 130–147, 2008.

[10] F. Sha and L. K. Saul, "Large margin training of continuous density hidden markov models," in *Automatic Speech and Speaker Recognition: Large Margin and Kernel Methods*, J. Keshet and S. Bengio, Eds. Wiley-Blackwell, 2009.

[11] D. Yu, L. Deng, X. He, and A. Acero, "Large-margin minimum classification error training for large-scale speech recognition tasks," in *Proceedings of the International Conference of Acoustic, Speech and Signal Processing (ICASSP-07)*, vol. 4, 2007, pp. 1137–1140.

[12] G. Saon and D. Povey, "Penalty function maximization for large margin HMM training," in *Proceedings of Interspeech-2008*, 2008.

[13] C. C. Cheng, F. Sha, and L. K. Saul, "A fast online algorithm for large margin training of continuous density hidden markov models," in *Proceedings of Interspeech-2009*, 2009.

[14] S. J. Young, "The general use of tying in phoneme-based HMM speech recognisers," in *Proceedings of the International Conference of Acoustic, Speech and Signal Processing (ICASSP-92)*, 1992, pp. 569–572.

[15] V. Digalakis and H. Murveit, "High-accuracy large-vocabulary speech recognition using mixture tying and consistency modeling," in *Proceedings of the workshop on Human Language Technology (HLT-94)*. Association for Computational Linguistics, 1994, pp. 313–318.

[16] L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," in *Proceedings of the International Conference of Acoustic, Speech and Signal Processing (ICASSP-86)*, Tokyo, 1986, pp. 49–52.

[17] B.-H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *IEEE Trans. Sig. Proceedings*, vol. 40, no. 12, pp. 3043–3054, 1992.

[18] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariahy, "Boosted MMI for model and feature-space discriminative training," in *Proceedings of the International Conference of Acoustic, Speech and Signal Processing (ICASSP-08)*, 2008, pp. 4057–4060.

[19] P. C. Woodland and D. Povey, "Large scale discriminative training for speech recognition," in *Proceedings of Automatic Speech Recognition (ASR-2000)*, 2000, pp. 7–16.

[20] C. C. Cheng, F. Sha, and L. K. Saul, "Matrix updates for perceptron training of continuous density hidden markov models," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[21] M. Collins, "Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms," in *Proceedings of Empirical Methods in Natural Language Processing (EMNLP-02)*, vol. 10, 2002, pp. 1–8.

[22] L. F. Lamel, R. H. Kassel, and S. Seneff, "Speech database development: design and analysis of the acoustic-phonetic corpus," in *Proceedings of the DARPA Speech Recognition Workshop*, L. S. Baumann, Ed., 1986, pp. 100–109.

[23] K. F. Lee and H. W. Hon, "Speaker-independent phone recognition using hidden markov models," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 37, pp. 1641–1648, 1988.