HIERARCHICAL LARGE-MARGIN GAUSSIAN MIXTURE MODELS FOR PHONETIC CLASSIFICATION

Hung-An Chang and James R. Glass

MIT Computer Science and Artificial Intelligence Laboratory
Cambridge, Massachusetts, 02139, USA
{hung_an,glass}@csail.mit.edu

ABSTRACT

In this paper we present a hierarchical large-margin Gaussian mixture modeling framework and evaluate it on the task of phonetic classification. A two-stage hierarchical classifier is trained by alternately updating parameters at different levels in the tree to maximize the joint margin of the overall classification. Since the loss function required in the training is convex to the parameter space the problem of spurious local minima is avoided. The model achieves good performance with fewer parameters than single-level classifiers. In the TIMIT benchmark task of context-independent phonetic classification, the proposed modeling scheme achieves a state-of-the-art phonetic classification error of 16.7% on the core test set. This is an absolute reduction of 1.6% from the best previously reported result on this task, and 4-5% lower than a variety of classifiers that have been recently examined on this task.

Index Terms— hierarchical classifier, committee classifier, large margin GMM, phonetic classification

1. INTRODUCTION

Over the years there has been much research devoted to the issue of acoustic modeling for automatic speech recognition. Topics of investigation have included feature representation, classifier structure, and training methods. It is fair to say that the most popular classifier structure in use today is the Gaussian mixture model (GMM), typically trained via maximum likelihood (ML) methods. To be sure, many alternative classifiers have been explored - especially those of a more discriminative nature such as neural-networks, conditional random-fields, support-vector machines, and other large-margin methods. In addition, many different discriminative training methods of GMMs have also been explored including maximum mutual information, minimum classification error [7], and more recently large-margin methods [2, 6, 14, 15]. One of the nice properties of large-margin-based methods is that the loss function of the training data is convex over the parameter space. Thus spurious local minima that may be encountered in other training scheme can be avoided.

In this paper we explore the use of a hierarchically structured large-margin GMM classifier. The use of hierarchies for acoustic modeling has not received as much attention in the literature, although there are some good results that have been achieved [1]. Hierarchies allow the potential for the classification problem to be divided into smaller sub-problems. Thus, there is the potential for a hierarchical classifier to be more robust since there are more training exemplars in the pooled classes. Hierarchies can also be used to partition a large feature vector into committees of smaller dimensionality classifiers, or to focus on particular acoustic measurements for a given class of sounds. Smaller dimensional classifiers offer the potential for more robust performance, and we have observed considerable benefit to such committees for ML-trained GMM classifiers in the past [5].

In this paper, we propose a hierarchical large margin GMM that incorporates the hierarchical classification with large margin training. In the remainder of the paper we first introduce the hierarchical GMM classifier in more detail and present large margin training within a hierarchical framework. We then describe experimental results on the benchmark TIMIT task of phonetic classification. Finally, we conclude and describe our future plans for research in this area.

2. HIERARCHICAL LARGE MARGIN GMMS

In this section, we first illustrate the hierarchical GMM classifier in more detail, and then describe the large margin training scheme for the hierarchical classifier. We focus here on a 2-level hierarchical classifier, however the scheme is generalizable to other kinds of hierarchies.

2.1. Hierarchical GMM Classifier

Consider a 2-level hierarchical GMM classifier H, illustrated in Figure 1, which can be constructed either by human knowledge or by automatic clustering algorithms. A leaf node c of H represents an individual class label (e.g., a phone) that H can output, and its parent node $s = S(c)$ represents the cluster where c belongs (e.g., a manner class). Each non-root node of H has a set of GMM parameters to model the distribution

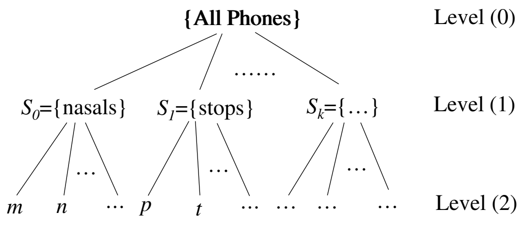

Fig. 1. Hierarchical classifier

of the feature vectors at that node. For convenience, we call class-level parameters for the GMM parameters of the nodes at the same level as c, and, similarly, cluster-level parameters for those at the same level as s.

To illustrate how H works, let us first consider the response of a class-level node c when a feature vector $\mathbf{x}$ is fed to H. Instead of looking at the log probability of the GMM directly, we look at the Mahalanobis distance returned by each mixture component of the model. Let μ_{cm} and $\mathbf{\Psi}_{cm}$ be the mean and inverse covariance for the m^{th} mixture component of c. Given $\mathbf{x}$, the mixture component returns a Mahalanobis distance

$$(\mathbf{x}-\mu_{\mathbf{cm}})^{\mathrm{T}}\mathbf{\Psi}_{cm}(\mathbf{x}-\mu_{\mathbf{cm}})+\theta_{cm}, \tag{1}$$

where θ_{cm} is a scalar offset that incorporates information of the mixture weight and the determinant of the mixture component. As in [2], we can reduce the expression of the distance in Eq. (1) into a compact form by introducing an extended feature vector $\mathbf{z}=[\mathbf{x}^{\mathrm{T}}\ 1]^{\mathrm{T}}\in\Re^{d+1}$ and an extended parameter matrix

$$\mathbf{\Phi_{cm}}=\left[\begin{array}{cc}\mathbf{\Psi}_{cm} & -\mathbf{\Psi}_{cm}\mu_{cm}\\ -\mu_{cm}^{\mathrm{T}}\mathbf{\Psi}_{cm} & \mu_{cm}^{\mathrm{T}}\mathbf{\Psi}_{cm}\mu_{cm}+\theta_{cm}\end{array}\right]. \tag{2}$$

By introducing $\mathbf{z}$ and $\mathbf{\Phi}_{cm}$, the distance in Eq. (1) can be expressed as

$$\mathtt{d}(\mathbf{\Phi}_{cm},\mathbf{z})=\mathbf{z}^{\mathrm{T}}\mathbf{\Phi}_{cm}\mathbf{z}. \tag{3}$$

Note that if we increment θ_{cm} by a constant, we can make the extended matrix $\mathbf{\Phi}_{cm}$ positive semi-definite. Combining the Mahalanobis distances of all mixture components of c, the overall distance of $\mathbf{x}$ to c can be computed by

$$\mathtt{D}(\mathbf{\Phi}_c,\mathbf{z})=-\log(\sum_m\exp(-\mathtt{d}(\mathbf{\Phi}_{cm},\mathbf{z}))), \tag{4}$$

where $\mathbf{\Phi}_c$ denotes the set of all extended parameter matrices $\{\mathbf{\Phi}_{cm}\}$ for all the mixture components of c. Similarly, for a cluster-level node s, we can have a set of parameter matrices $\mathbf{\Theta}_s$ for the cluster-level model, and when given a feature $\mathbf{x}$, the model of s returns distance

$$\mathtt{D}(\mathbf{\Theta}_s,\mathbf{z})=-\log(\sum_k\exp(-\mathtt{d}(\mathbf{\Theta}_{sk},\mathbf{z}))). \tag{5}$$

With the distances computed by the two levels of nodes, H outputs a predicted label $\widehat{y}$ for $\mathbf{z}$ by the following criterion:

$$\widehat{y}=\arg\min_c\{w_C\mathtt{D}(\mathbf{\Phi}_c,\mathbf{z})+w_S\mathtt{D}(\mathbf{\Theta}_{S(c)},\mathbf{z})\}, \tag{6}$$

where w_C and w_S are the relative weights that reflect how H trusts the information from the two levels, respectively, and $S(c)$ is the parent of c. Generally, w_C and w_S can be determined by cross-validation with held-out training data. Note that since the computation for the classification involves only basic calculations of GMMs plus fixed amounts of additions and multiplies, the overall complexity of the hierarchical classifier is similar to conventional GMM classifiers.

2.2. Hierarchical Large Margin Training

Here we present the parameter training scheme for the hierarchical classifier given a set of labeled training examples $\{(\mathbf{x}_n,y_n)\}_{n=1}^N$, where $\mathbf{x}_n\in\Re^d$ and $y_n\in\{1,2,\ldots,C\}$. Given the class label y_n, we can get the cluster label $s_n=S(y_n)$ by the hierarchical tree. We use $\{\mathbf{\Phi}_c^{\mathrm{ML}}\}$ to denote the maximum-likelihood (ML) class-level parameters before the large margin training, and similarly use $\{\mathbf{\Theta}_s^{\mathrm{ML}}\}$ for the cluster-level parameters. Also, we use $\mathbf{z}_n$ to denote the extended vector of $\mathbf{x}_n$.

As in [2], we seek model parameters such that each training example is correctly classified by a large margin. For each $\mathbf{x}_n$ (or equivalently $\mathbf{z}_n$), consider the distance

$$w_C\mathtt{d}(\mathbf{\Phi}_{y_nm_n},\mathbf{z}_n)+w_S\mathtt{d}(\mathbf{\Theta}_{s_nk_n},\mathbf{z}_n), \tag{7}$$

where $m_n=\arg\min_m\{\mathbf{z}_n^{\mathrm{T}}\mathbf{\Phi}_{y_nm}^{\mathrm{ML}}\mathbf{z}_n\}$ is the mixture component of y_n that is closest to $\mathbf{z}_n$ according to the initial model $\mathbf{\Phi}_{y_n}^{\mathrm{ML}}$ and $k_n=\{\arg\min_k\mathbf{z}_n^{\mathrm{T}}\mathbf{\Theta}_{s_nk}^{\mathrm{ML}}\mathbf{z}_n\}$. Note that the distance in Eq. (7) is a lower bound of the distance of the correct label used in classification. If we can have

$$\forall c\neq y_n,\quad \begin{array}{c}w_C\mathtt{D}(\mathbf{\Phi}_c,\mathbf{z}_n))+w_S\mathtt{D}(\mathbf{\Theta}_{S(c)},\mathbf{z}_n)\geq\\ 1+w_C\mathtt{d}(\mathbf{\Phi}_{y_nm_n},\mathbf{z}_n)+w_S\mathtt{d}(\mathbf{\Theta}_{s_nk_n},\mathbf{z}_n)\end{array}, \tag{8}$$

then we can guarantee $\mathbf{z}_n$ is correctly classified by at least 1 unit margin. Any violation of the criterion in Eq. (8) is considered a loss, and we can compute the loss for each training example by

$$\begin{array}{r}\ell_n=\sum_c[1+w_C(\mathtt{d}(\mathbf{\Phi}_{y_nm_n},\mathbf{z}_n)-\mathtt{D}(\mathbf{\Phi}_c,\mathbf{z}_n))\\ +w_S(\mathtt{d}(\mathbf{\Theta}_{s_nk_n},\mathbf{z}_n)-\mathtt{D}(\mathbf{\Theta}_{S(c)},\mathbf{z}_n))]_+\end{array}, \tag{9}$$

where the function $[f]_+=\max(0,f)$. By this definition, ℓ_n is a convex function over the parameter space of $\mathbf{\Phi}$ and $\mathbf{\Theta}$.

We also use a weighted sum of ℓ_n to be the final loss function we try to minimize; that is

$$\mathfrak{L}=\sum_n w_n\ell_n, \tag{10}$$

where ℓ_n is computed by Eq. (9) and the weight w_n is chosen such that correcting each training sample contributes roughly the same amount of reduction in the loss function. More specifically, as in [2], we choose $w_n=\min(1,\frac{1}{\ell_n^{ML}})$, where ℓ_n^{ML} is the loss of the n^{th} example under the initial ML model.

By setting w_n in this way, we can effectively prevent the outliers in the training example from seriously affecting the result. Because the loss function in Eq. (10) is convex to the positive semi-definite parameter matrices $\mathbf{\Phi}$ and $\mathbf{\Theta}$, we can use a convex optimization algorithm such as conjugate gradient (CG)[9] or other positive semi-definitive programming methods to find the optimal set of parameters[10].

Since we have two levels of parameters, we can implement the training by the following method. We first fix cluster level matrices $\mathbf{\Theta}$ and adjust $\mathbf{\Phi}$ and $\mathfrak{L}$ using CG. After a certain number of iterations, we then switch to adjust $\mathbf{\Theta}$ while fixing $\mathbf{\Phi}$. We repeat these two steps for several rounds until the CG terminates automatically, or a maximum number of iterations is reached. The idea of this training scheme is similar to that of a turbo code in that we use the output of the first level in the hierarchy to help optimize the second level, and vice versa. By doing this, we can reduce the original optimization problem into 2 sub-problems with fewer parameters to update and thus the algorithm can run more efficiently. In our TIMIT phonetic classification experiments we set $t_1 = 50$, $t_2 = 60$, and $r = 3$.

Algorithm 1 Turbo Training

```
1: Fix $\mathbf{\Theta}$, run CG on $\mathbf{\Phi}$ for $t_1$ iterations to minimize $\mathfrak{L}$.
2: Fix $\mathbf{\Phi}$, run CG on $\mathbf{\Theta}$ for $t_2$ iterations to minimize $\mathfrak{L}$.
3: Repeat 1 and 2 until CG stops or $r$ rounds have reached.
4: Use held-out training data to choose the final models.
```

Note that the margin in the training is not necessarily fixed. We can set up different margin constraints by scaling the parameter matrices $\mathbf{\Phi}$ and $\mathbf{\Theta}$ with a factor α before computing ℓ_n. Different values of α can have a large impact on the resulting models. Effectively, a smaller α results in a larger margin. This will potentially make more training samples have a positive loss and thus make more training examples considered during training. In general, more samples being considered in the optimization can result in a more robust decision boundary so that the resulting model will be more generalizable to unseen data. However, if we choose a very small α, the large margin training will include many examples that may not be very informative for selecting a good decision boundary and will therefore limit the gain of the large margin training. Thus, choosing a good scaling factor is important and we will discuss this issue in the next section.

3. EXPERIMENTS

In this section, we present the classification results of the hierarchical large margin GMMs on the well-defined TIMIT benchmark task of context independent phonetic classification [13]. In addition to exploring single hierarchical classifiers, we also explored the use of committee-based classifiers to improve performance [5].

Method	Feature	Error Rate
Hierarchical GMM[1]	Seg	21.0%
Hidden CRF[3]	Frame	21.7%
Large Margin GMM[2]	Frame	21.1%
RLS2[4]	Seg	20.9%

Table 1. Recent reported results on TIMIT core test set. Feature type refers to segmental (1 vector/phone) or frame-based.

3.1. Corpus Setup

In our experiments we used the standard NIST training set (462 speakers, 3696 utterances, 140225 tokens) for training, and standard core test set (24 speakers, 192 utterances, 7215 tokens) for testing. In addition, we also used the standard development set (50 speakers, 400 utterances, 15056 tokens) to decide the relative weights of the two level models in the hierarchy, to provide early stopping of the training, and to tune the weight of the phone prior. The development set was also used to tune the optimal value of the margin scaling factor α that was used during training.

The standard 61 TIMIT phone labels were reduced into 48 classes as in [8]. When evaluating the models, we further mapped the labels into the commonly used 39 classes [1]-[4] to calculate the classification error rate. As in commonly done, we also ignored glottal stops ($/q/$) for both training and testing. The error rate of the reduced 39-class classification on the core test set of TIMIT is a well-defined benchmark problem. Table 1 lists the results of some recently reported experiments for this task.

3.2. Features

In the experiments, we trained models for the eight different segmental feature measurements (i.e., one vector per phone) that were used in [5]. The eight different measurements, S1-S8, are summarized in Table 2. They differ primarily in a) the duration of the Hamming window used to compute the short-time Fourier transform, b) the number of Mel-frequency Cepstral Coefficients (MFCCs) or perceptual linear prediction (PLP) coefficients used and c) whether the coefficients were consolidated via a temporal basis function (that extended 30ms beyond the segment boundaries) of either averages or cosine transforms. Each feature vector also included log duration. The number of dimensions of each type of feature is determined by number of spectral coefficients and the number of temporal basis functions. For example, S1 has $5*12+1 = 61$ dimensions.

3.3. Baselines

For the classification experiments, we built the 2-level hierarchy by clustering the phone class into nine clusters according to their broad manner of articulation. The nine clusters

	# Dims	Window [ms]	Spectral Representation	Temporal Basis
S1	61	10	12MFCC	5 avg
S2	61	30	12MFCC	5 avg
S3	61	10	12MFCC	5 cos
S4	61	30	12MFCC	5 cos
S5	64	10	9MFCC	7 cos
S6	61	30	15MFCC	4 cos
S7	61	20	12PLPCC	5 avg
S8	61	20	12PLPCC	5 cos

Table 2. Summary of features used for experiments.

Set	Gauss	2-mix	4-mix	H(1,2)	H(2,4)
Dev	24.8%	23.8%	23.5%	24.7%	23.8%
Test	25.2%	24.4%	24.1%	25.2%	24.3%

Table 3. Error rates of the ML GMM classifiers.

are stops, nasals, strong fricatives, weak fricatives, high vowels, low vowels, short vowels, semi-vowels, and closures (including silences). For each of the eight features, we trained 5 kinds of ML baseline models: "Gauss","2-mix", "4-mix", "H(1,2)", and "H(2,4)". "Gauss" refers to a single full covariance Gaussian model, while "2-mix" and "4-mix" represent GMMs with two and four Gaussian components respectively. "H(1,2)" is a hierarchical model using one Gaussian at the class-level model and two Gaussian components for the cluster-level model; "H(2,4)" is defined similarly. The GMMs were trained by the cross-validation EM (CV-EM) algorithm [11] and selected by the development set. (Pick the one with lower error rate among two independent trails.) For "H(1,2)" and "H(2,4)", we also used the development set to find a proper set of relative weights w_C and w_S between the two levels of the hierarchy.

Table 3 lists the average error rates of the ML models on the development and core test set when trained on the eight different feature sets, S1-S8. From the table, we can see that, on average, the performance of "H(1,2)" is close to "Gauss" and that of "H(2,4)" is close to "2-mix", showing that ML training does not derive much benefit from the hierarchical framework. Although the data is not shown in the table, we also observed that, for two of the feature sets, the "4-mix" models performed worse than their corresponding "2-mix" models, showing that in some cases the models were overfitting the training data.

3.4. Large margin models

3.4.1. Single classifier results

In this section we present the classification results of the models for each type of feature after the large margin training. The

Set	Gauss	2-mix	4-mix	H(1,2)	H(2,4)
Dev	19.2%	18.7%	18.8%	18.8%	18.5%
Test	20.6%	20.0%	20.0%	19.9%	19.6%

Table 4. Error rates of the large margin GMM classifiers.

large margin models "LM Gauss", "LM 2-mix", and "LM 4-mix" are trained as in [2], while "LM H(1,2)" and "LM H(2,4)" are trained by the scheme presented in the previous section.

As mentioned previously, the margin scaling factor α can significantly affect the performance of the models. To illustrate how the model performances vary according to α, we sample several values of α and plot the average error rate of the eight features on the development set in Figure 2. As the figure shows, the error rates decrease as α is reduced from 0.25 to 0.05, and increase as α gets smaller than 0.05. The trend of the curves are as discussed in Section 2. Another interesting observation is that the more complex the model is, the greater the variation in classification performance; indicating that finding a good value of α becomes important as the model become more complex.

Table 4 shows the average results of the models for the eight feature sets on the development and core test set, respectively, under $\alpha = 0.05$. From the table we can see that although "LM 4-mix" has almost twice the number of parameters as "LM 2-mix", the performances of the two kinds of models are on average quite similar. This shows that simply increasing the number of mixtures may not necessarily improve the overall performance, since the model may over-fit the training data. On the other hand,"LM H(1,2)" and"LM H(2,4)" achieve better performance than the other three models on average, showing that the hierarchical models are perhaps generalizing better to unseen data. To see whether the proposed modeling scheme has significant improvement over the current state of art, we compared the outputs of "LM H(2,4)" with that of RLS2 model [4] and conducted a McNemar significance test [12]. Six out of the eight models were significantly different at the 0.001 level. This also includes the model trained with feature set S2, which was also used for the RLS2 experiments.

3.4.2. Committee classifiers

In addition to performing classification with a single feature vector, we can also create a committee-based classifiers that combines information provided by the different feature sets. As in [5], our committee-based classifier combined the outputs of the individual classifiers for S1-S8 by summing their log posterior probabilities. The performances of the committee-based classifier was also affected by the margin scaling factor α. Figure 3 shows the performances of the committee-based classifiers on the development set under different value of α.

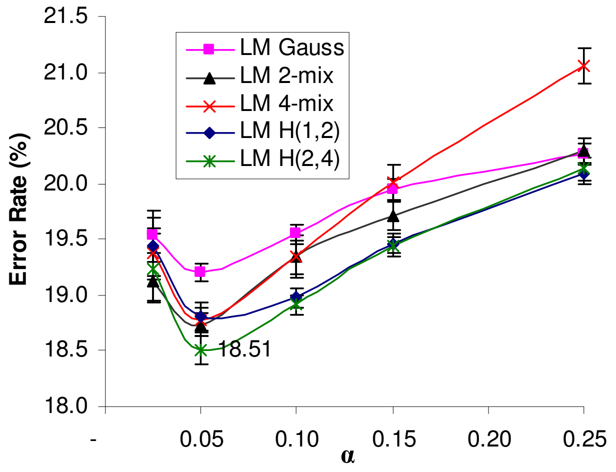

Fig. 2. Average error rate on the development set. Error bars show 0.25 standard deviation across feature sets.

Set	Gauss	2-mix	4-mix	H(1,2)	H(2,4)
Dev	17.0%	16.2%	16.1%	16.5%	15.9%
Test	17.8%	17.1%	17.1%	17.2%	16.8%

Table 5. Error rates of committee classifiers.

The detailed performances of the committee classifiers on the development and test sets (using $\alpha = 0.1$) are listed in Table 5. As in the earlier large-margin experiments, the "H(2,4)" model yields the best result.

We found it interesting to observe that for all five types of classifiers we explored the optimal value of α for an individual classifier did not result in the best committee classifier. The optimal value of the committee based classifiers tended to be consistently slightly larger than that of individual classifiers. One possible explanation for this observation could be that the diversity of the individual classifiers may tend to decrease as α becomes smaller, since the overlap of the training examples used in the large margin training would tend to become larger. As a result, although each individual classifier became more accurate, they became less complementary of each other and thus the overall committee was not as effective as the one with a set of more diverse but reasonably accurate committee members.

3.4.3. *Heuristic selection of α*

In the previous experiments we used a brute force search on the development set to find a good value of α. Although this method was effective, it was also time consuming in that we have to first have the trained models before we can evaluate the performances on the development set. It would be much preferable if we could find a suitable value of α before training, especially for the case of large vocabulary continuous speech recognition where discriminative training may take a very long time.

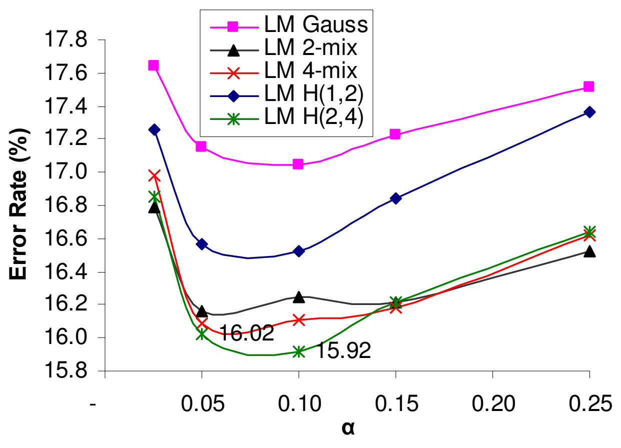

Fig. 3. Average error rate on the development set.

We explored a heuristic method for finding α which was inspired from the observation that, for a training token with positive loss, the value of the loss also has a convex shape variation according to α. To explain this, let us consider a training example n with positive loss. We call such kind of training examples "effective" since only such examples would be considered in the large margin training. For convenience, we shorten the expression of the loss by $\ell_n = \sum_{c \neq y_n} [1 + \alpha\Delta_c]_+$.

A small α can have a two-sided effect on ℓ_n. On the one hand, for a strong competing class c_1 with $\Delta_{c_1} > 0$, small α can make $[1 + \alpha\Delta_{c_1}]_+$ small, and in this sense, may decrease the loss value. On the other hand, a small α may also make a weak competing class c_2 with $\Delta_{c_2} \ll 0$ contribute to the loss by making $1 + \alpha\Delta_{c_2} > 0$, and may also increase ℓ_n. Because the loss is piece-wise linear to α, there exists an $\widehat{\alpha}$ such that ℓ_n can be minimized.

In other words, there exists a certain value of α that can minimize the loss for n, and that value can potentially be a good choice for n since it balances the two effects: trying to increase margin as much as possible (or, equivalently, choosing α as small as possible) while not letting too many weak competing classes disrupt the training. Following this idea, if we pick a value $\widehat{\alpha}$ that minimize the average loss of the all effective examples, that value may also be a suitable choice of α for the whole training set.

We applied this heuristic approach to select α for "LM H(1,2)" and "LM H(2,4)". The performances of the resulting model for all eight feature sets is listed in Table 6. The "com" in the table refers to committee classifier. Both of the two sets of models have performances very close to the models with $\alpha = 0.05$, which demonstrates the effectiveness of the heuristic method. Note that this table shows the best overall result for any single feature set obtains an error rate of 18.7%, while the best overall committee-based classifier obtains an

	LM H(1,2)			LM H(2,4)		
	dev	test	α	dev	test	α
S1	18.9%	20.0%	0.070	19.1%	20.4%	0.070
S2	18.9%	19.8%	0.070	18.3%	19.4%	0.072
S3	18.3%	19.6%	0.063	17.9%	19.7%	0.069
S4	18.3%	19.4%	0.067	18.1%	**18.7%**	0.069
S5	18.7%	19.7%	0.064	18.6%	19.6%	0.062
S6	19.1%	20.2%	0.068	18.7%	20.4%	0.069
S7	19.6%	21.0%	0.076	19.2%	20.6%	0.071
S8	18.7%	19.9%	0.067	18.7%	19.8%	0.071
avg	18.8%	20.0%		18.6%	19.8%	
com	16.6%	17.2%		16.0%	**16.7%**	

Table 6. Error rates of classifiers with pre-determined α.

error rate of 16.7%.

4. CONCLUSION AND FUTURE WORKS

In this paper we have incorporated large margin GMM training into a hierarchical classification framework. The resulting classifier obtains excellent performance on the task of TIMIT phonetic classification, achieving 18.71% error rate in a single classifier case and 16.74% in the case of committee-based classification. Because the proposed method can incorporate additional phonetic information, it can achieve excellent performance while using fewer parameters than any other state-of-the-art technique. By using fewer parameters, the proposed model not only requires less computations but also can avoid over-fitting when confronted with data sparsity problems.

In the future, we plan to extend this work to the task of phonetic recognition to see how much gain in the classification can be transferred to phonetic recognition. Our prior experience in this area indicates that improvements in classification typically carry over to reduced substitution errors in recognition tasks [5]. We would also like to conduct experiments on context-dependent phone models for word-based recognition. Generally, the data sparsity problem becomes more severe in context-dependent modeling, and our hope is that the hierarchical large margin modeling method should be effective for this problem as well.

Acknowledgements

The authors are grateful to Fei Sha for providing much helpful advice on large margin GMM training issues. This work was supported by Taiwan Merit Scholarships (number NSC-095-SAF-I-564-040-TMS) and by the T-Party Project, a joint research program between MIT and Quanta Computer Inc., Taiwan.

5. REFERENCES

[1] A.K. Halberstadt and J.R. Glass, "Heterogeneous acoustic measurements for phonetic classification," *Proceedings of Eurospeech*, pp. 401–404, 1997.

[2] F. Sha and L.K. Saul, "Large margin gaussian mixture modeling for phonetic classification and recognition," *Proceedings of ICASSP*, pp. 265–268, 2006.

[3] A. Gunawardana, M. Mahajan, A. Acero, and J.C. Platt, "Hidden conditional random fields for phone classification," *Proceedings of Eurospeech*, 2005.

[4] R. Rifkin, K. Schutte, M. Saad, J. Bouvire, and J.R. Glass, "Noise robust phonetic classification with linear regularized least squares and second-order features," *Proceedings of ICASSP*, pp. 881–884, 2007.

[5] A.K. Halberstadt and J.R. Glass, "Heterogeneous acoustic measurements and multiple classifiers for speech recognition," *Proceedings of ICSLP*, 1998.

[6] F. Sha and L.K. Saul, "Comparison of large margin training to other discriminative methods for phonetic recoginition by hidden markov models," *Proceedings of ICASSP*, pp. 313–316, 2007.

[7] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 5, no. 3, pp. 257–265, 1997.

[8] K.F. Lee and H.W. Hon, "Speaker-independent phone recognition using hidden markov models," *IEEE Transactions on Acoustic, Speech, and Signal Processing*, vol. 37, no. 11, pp. 1641–1648, 1988.

[9] J.R. Schewchuk, "An introduction to conjugate gradient method without the agonizing pain," *C.M.U.*, 1994.

[10] L.Vandenberghe and S.P. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, March 1996.

[11] T. Shinozaki and M. Ostendorf, "Cross-validation EM training for robust parameter estimation," *Proceedings of ICASSP*, pp. 437–440, 2007.

[12] L. Gillick and S.J. Cox, "Some statistical issues in the comparison of speech recognition algorithms," *Proceedings of ICASSP*, pp. 532–535, 1989.

[13] John S. Garofolo et al, "TIMIT acoustic-phonetic continuous speech corpus," *Linguistic Data Consortium*, 1993.

[14] X. Li, H. Jiang, and C. Liu, "Large margin HMMs for speech recognition," *Proceedings of ICASSP*, pp. 513–516, 2005.

[15] D. Yu, L. Deng, X. He, and A. Acero, "Large-margin minimum classification error training for large-scale speech recognition tasks," *Proceedings of ICASSP*, pp. 1137–1140, 2007.