SELECTION, PARAMETER ESTIMATION, AND DISCRIMINATIVE TRAINING OF HIDDEN MARKOV MODELS FOR GENERAL AUDIO MODELING

Manuel J. Reyes-Gomez and Daniel P.W. Ellis

Dept. of Electrical Engineering, Columbia University, New York, NY 10027

ABSTRACT

Hidden Markov Models (HMMs) permit a natural and flexible way to model time-sequential data. The ease of concatenation and time-warping algorithms implementation on HMMs suit them very well for segmentation and content based audio classification applications, as is clear from their extended and successful use on speech recognition applications.

Speech has a natural basic unit, the phone, which normally delimits the number of models to one per phone. Moreover, knowledge of the speech structure facilitates the choice of the model parameters. When modeling generic audio, on other hand, the lack of a natural basic unit, and the absence of a clear structure, make the selection and the parameter estimation of an optimal set of HMMs difficult.

In this paper we present different approaches to select and estimate the HMM parameters of a set of representative generic audio classes. We compare these approaches in the context of a content-based classification application using the MuscleFish database.

The models are first found through frame clustering or by traditional EM techniques under some specific selection criteria, such as the Bayesian Information Criterion. Further discriminative training of the initial models considerably improve their performance in the content-based classification task, obtaining results comparable with the ones obtained, for the same task, by inherently discriminative classification methods, such as support vector machines, while preserving the intrinsic flexibility of HMMs.

1. INTRODUCTION

The time sequential nature of HMMs enable the effective implementation of time warping algorithms. This feature and the HMM's ease of concatenation permit the use of these models to effectively segment and decode streams of audio. This is evidently shown in their extended and successful use in speech recognition.

Speech has a clear modular structure which is used to limit the variables involved in the modeling. Typically, a single model is used to represent each of the phones in the dictionary [8]. Moreover each model is generally represented by a left-to-right HMM with 3 or 4 states [8].

For the case of general audio, however, there is no basic unit like a phone, nor an obvious basis for the choice of the number of states or the topology. At the same time, the choice of these parameters has a tremendous impact on the performance of any system attempting to model generic audio, regardless of the application.

We can view any audio stream as a sequence of representative "types" of sounds. A particular sequence of these "types" of sounds could have an associated meaning, e.g., a sequence of shooting sounds and screaming sounds implies a shooting scene. The choice of the "types" of sounds, and the detail to be discriminated within each one, is very much application dependent, just as the choice of the phone set to be used in a speech recognizer is language and vocabulary dependent.

In this paper we present the implementation of models for the 16 "types" of sounds contained in the MuscleFish database [7]. We present different approaches to choose and estimate the model's parameters. Results in the context of a content-based audio classification application are also presented. The models are first found through frame clustering or by traditional EM techniques under some specific selection criteria, such as the Bayesian Information Criteria. Further discriminative training of the initial models considerably improves their performance in the content-based classification task, obtaining results comparable with the ones obtained by inherently discriminative classification methods, such as support vector machines.

Hidden Markov Models have been used before to model environmental audio files in [9] [2], but no generalization was attempted since a single HMM was used to model each file. In [1], HMM models for audio classes are found through minimum entropy training. Even though this technique can potentially discover the structure of the model, it requires an initial overparametrized model, which implies an idea of the ideal number of states.

The MuscleFish database has been used in several previous studies offer content-based audio classification. It was first introduced in [7], where a normalized Euclidean distance and the nearest neighbor classifier are used to classify the input sound into one of the classes in the database. In [3], a multi-class classifier is implemented by combining several two-class support vector machines. They have reported the best performance in this particular task, however support vector machines classifiers cannot easily decode streams of audio, since they normally require some presegmentation of the data, making its application in other audio processing applications difficult. Our models, after discriminative training, achieve similar results to the ones obtained in the latter work, but are also applicable to non-homogeneous streams.

In section 2, the objective of the work described in this paper is presented. Section 3 proposes different approaches to select and estimate the HMM parameters when used to model generic audio. Section 4 introduces the further discriminative training of the models found in section 2, and section 5 presents results using the models obtained from sections 3 and 4, in the context of a content-based audio classifier. Finally, in section 6, the conclusions and future work are discussed.

2. OBJECTIVE: CLASS MODELS

As mentioned above, the number and kind of representative "types" of sounds is application dependent. In our case it was con-

Submitted to ICME-2003, Baltimore

strained by the kind of files in the MuscleFish Database [7], which consists of approx. 420 audio files divided into 16 audio classes. We divided the database in a training and a test subsets of roughly equal size.

Our objective is to find spectral and temporal patterns between the (training) files in each class and then model them using a (Hidden Markov) Class Model. If these patterns are correctly identified and modeled, a segment in an audio stream containing these classes of sounds could be successfully identified. For example, we want to be able (as we did in the application discussed in section 3) to identify a segment containing ducks quacking as an "animal" sound using a model trained with chicken, dog, pig and horse sounds.

To represent each class file, we used the first 18 cepstral coefficients computed every 10ms with a window size of 25ms. Each feature is then normalized to zero mean and unit variance.

3. CLASS HMM PARAMETERS SELECTION AND ESTIMATION

Hidden Markov Models have three principal parameters.

1. Number of states.
2. State Likelihood representation.
3. Transition Matrix (topology).

The choice of the number of states is particularly important, since we want to have enough states to differentiate the different representative acoustical patterns present in a class file. But at the same time, we wish to avoid overfitting the data through a very detailed model, which could be the case when too many states are used.

We used two different approaches with several variants to estimate these parameters.

1. Clustering
2. GMM-EM, with several model selection criteria.

3.1. Clustering

A clustering algorithm is used to find the different frame clusters with similar acoustic properties between the files on the same class. Each cluster corresponds to a state; the likelihood representation and the transition matrix are then defined using the frames within each cluster. The main problem is that we do not have any prior knowledge of how many different clusters will be in any given class file. We used a K-variable version of the K-means algorithm (a modified version of the algorithm used in [9]).

K-Variable, K-means algorithm: This is a greedy algorithm where clusters are successively added to account for points that lie beyond some threshold distance from any existing cluster.

```
Initialize the algorithm with one cluster
Make the frame with the highest norm the center of cluster 1.
For each remaining frame $x_i$ in the file
    find d = min ($d_{ij}$($x_i$,$c_j$));
    if d $< T_1$, make $x_i$ member of cluster j.
    if d $> T_2$, make $x_i$ the center of a new cluster.
Repeat until $V = \sum_j \sum_k (xjk - c_k)^2$ (the within cluster variance) reaches a minimum.
After convergence, make all the remaining unclassified frames members of their closest cluster.
```

Here, d_{ij} is the Euclidean distance, c_j is the center of cluster j. $T_1 = m - C \cdot s$ and $T_2 = m + C \cdot s$; m and s are the mean and standard deviation of the distances between any pair of frames within the class file. C is a parameter in the range [0.5 1.5]. The number of clusters obtained is closely correlated with the value of C, the lower the value of C, the more clusters are present in the final partition.

Clusters with fewer than 10 frames are eliminated since they are considered not representative of the class; they are considered to be produced by singularities in a particular file. Table 2 shows the actual number of clusters obtained for each class in the MuscleFish database under various methods. The states likelihoods are estimated using a Gaussian Mixtures with diagonal covariance matrices. The number of Gaussian mixture components m depends on the number of frames in each cluster according to the rule in table 1. The parameters of the Gaussian mixtures are calculated via the Expectation Maximization algorithm.

Frames per cluster	# Gaussians	Frames per cluster	# Gaussians
0-49	1	100-149	3
50-99	2	150-Up	4

Table 1. Number of mixtures used in a cluster frame distribution per number of frames

The clustering algorithm generates in each training class from a given class frame membership sequences like 1144422223333, where the numbers represent the cluster membership of the frame. The transition probabilities are estimated by counting the transition between adjacent frames, by the following formula:

$$P(i,j) = \frac{n_{i,j}}{n_i} \tag{1}$$

where $n_{i,j}$ is the number of transitions from state i to state j and n_i is the total number of transitions out of state i.

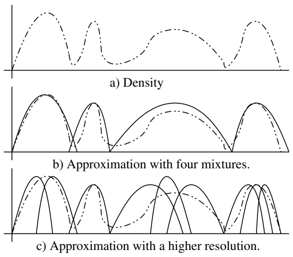
3.2. GMM-EM algorithm

Conventional HMM parameter estimation through the EM algorithm requires a previous knowledge of the number of states and a good initialization of the transition matrix. Results for different initializations of the transition matrix are discussed in section 4. We use three different criteria to select the number of states or to choose a model out of a pool of models:

1. Low Entropy Criterion.
2. Low State Occupancy Criterion.
3. Bayesian Information Criterion.

Low Entropy Criterion: We use mixtures of Gaussians to estimate the number of states and the initial transition matrix for each class model.

```
We start a single Gaussian ($m = 1$). Make Stop = 0
While Stop = 0
    Increment the number of Gaussians $m = m + 1$;
    Find GMM parameters for all the feature vectors from files within the same class via the EM algorithm.
    Assign each frame to the Gaussian with the highest likelihood.
    Find transition probabilities counting transitions between Gaussians in successive frames.
    If any self-loop probability is smaller than $\alpha$, set Stop = 1
```



a) Density

b) Approximation with four mixtures.

c) Approximation with a higher resolution.

Fig. 1. Density approximation by mixture of Gaussians. If too many Gaussians are used, the correspondence between Gaussians and modes in the data is lost.

```
End
    Set the number of states equal to $m-1$, initialize the HMM transition matrix to the matrix associated with the $m-1$ Gaussians.
```

Once we obtain the number of states and the transition matrix, the actual Gaussians are discarded and the values of the mean and diagonal variance for each state are initialized to the global values for the feature vectors of all files within the given class. Then the EM algorithm is applied to find the model parameters through a maximum-likelihood criteria, using the HTK framework [8]. During training, the number of Gaussians per state is determined by the number of frames associated with a given state under a forced alignment, according to table 1. The number of states found using this approach with α = 0.5 can be observed in table 2, which also gives the number of states found using only one Gaussians per state.

Class	Variable K-Means	BIC	Low Entropy	Low State Occu.
Altotrombone	6	15 / 33	4 / 4	7 / 12
Animals	7	35 / 38	8 / 8	10 / 10
Bells	3	27 / 27	3 / 3	7 / 9
Crowds	6	22 / 32	3 / 3	10 / 19
Cellobowed	3	15 / 39	9 / 9	13 / 19
Female	3	9 / 17	7 / 7	11 / 11
Laughter	5	19 / 29	8 / 8	21 / 25
Machines	8	32 / 34	4 / 4	16 / 23
Male	3	11 / 19	3 / 3	4 / 10
Oboe	21	37 / 37	4 / 4	17 / 21
Percussion	24	36 / 37	10 / 10	23 / 40
Telephone	3	24 / 31	3 / 3	11 / 14
Tubularbells	6	30 / 37	3 / 3	10 / 17
Violinbowed	27	30 / 38	8 / 8	12 / 27
Violinpizz	4	20 / 16	8 / 8	11 / 19
Water	8	34 / 38	8 / 8	10 / 42
Performance	84.6 / 88.9	74.0 / 77.8	89.9 / 82.2	83.6 / 84.1

Table 2. Number of states per class under each approach. The first number represents the number of states obtained when mixtures of Gaussian are used to estimate the emission probability. The second number (after the slash) is the number of states obtained when single Gaussians are used. The last row represents the performance of the models for multiple Gaussian and single Gaussians, except for the Variable K-means where the second number is performance when 4 models are used for each class.

The idea behind the restriction in the self-loop probabilities is that frames are *spatially correlated*, meaning that adjacent frames have similar feature values and by association have a higher probability of transitioning to the same state, so we can use this to control the resolution (number of Gaussians) of the GMM and avoid overfitting. For the one-dimensional case portrayed in figure 1, part b), uses four mixtures to approximate the desired distribution. There, similar values of x have a high probability to belong to the same Gaussian. In part c), where the GMM has a higher resolution this no longer holds.

Low State Occupancy Criterion: The same basic algorithm is used to initialize the HMM, but with a different stopping criterion, based on a minimum number of frames being assigned to each state:

```
    For each class model, we start with a single Gaussian ($m = 1$).
Set Stop = 0.
While Stop = 0
    Find model transition matrix using the previous algorithm.
    Estimate parameters through EM algorithm.
    During training, if the number of frames associated with any state is below a certain threshold $\tau$, set Stop = 1.
End
```

The final completely-trained model (i.e at $m-1$) is taken as the class model. The number of states found using this approach, for τ = 10, can be observed in table 2.

Bayesian Information Criterion, BIC: A pool of 50 models for each class model is found, by setting $m = 1 \cdots 50$ in the previous algorithms without any restriction. Each class is represented by the model with the largest BIC criterion [6], defined as:

$$\theta^* = \underset{\theta_i}{\text{argmax}} \; (\sum_{n=1}^{N} (logP(X_n \mid \theta_i)) - \frac{d_i}{2} logN); \qquad (2)$$

where θ_i is a model parameter set with d_i degrees of freedom, and X_n is a data set with N elements.

4. DISCRIMINATIVE TRAINING

Previous work in speech recognition [5] has shown that the combined use of generative and discriminative training can substantially enhance the performance of acoustic models. Thus, we further apply discriminative training to the models previously obtained. We define an objective function $F(X_i, \theta)$ and a misclassification function $d_C(X_i, \theta)$ as:

$$F(X_i, \theta) = \frac{1}{1 + e^{-d_C(X_i, \theta)}} \qquad (3)$$

$$d_C(X_i, \theta) = \frac{-log(P(X_i \mid \theta_{C_i})) + log(P(X_i \mid \theta_{I_i}))}{\kappa} \qquad (4)$$

where θ_{C_i} are the parameters of the correct class C_i, and θ_{I_i} are the parameters of the incorrect class I_i with the highest log-likelihood for observation X_i. A similar approach was used in [4] for phonetic classification.

Discriminative training is done by taking the derivatives of $d_C(X_i, \theta)$ with respect to θ_{C_i} and θ_{I_i} and applying a gradient descent algorithm for all the files X_i in the training set. Starting with a single mixture state model, we limit the further discriminative training to the means only. The algorithm works as follows:

```
For number of iterations.
    For all files $X_i$ in the training set.
        Update mean $\mu_k$ for $k = I_i$ or $C_i$ by:
```

$$\mu_k^s = \mu_k^s \pm \frac{\eta}{\kappa} F(X_i, \theta)(F(X_i, \theta) - 1)\Sigma_k \sum_{j \in T_k} (x_i^j - \mu_k^s) \qquad (5)$$

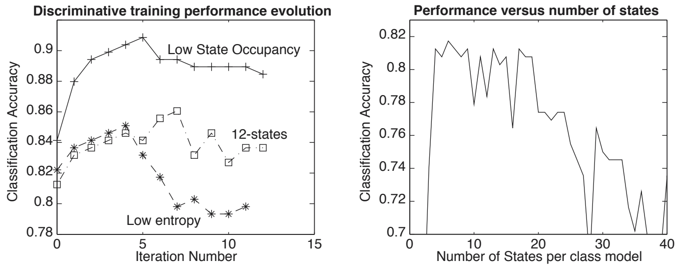


Fig. 2. Left: Classification accuracy of single mixture models during discriminative training iterations. Right: Classification performance for sets of models with the same number of states.

where the update is in the positive sense for the correct class, and negative for the incorrect class. T_k are the time indexes for the frames emitted by state s on model k.

5. CONTENT-BASED APPLICATION RESULTS

The class models found on the previous sections were used in a content-based classification application. The database is divided into two sets. The training set was used to estimate the class models, which were then used to classify each test example according to the model with the Viterbi alignment with the greatest likelihood. The classification accuracy for the different sets of models are summarized in the last row of table 2.

Different values for the C parameter in the variable K-means approach result in different partitions of the class files. The performance of the models obtained via this approach for C values in the range [0.5,1.5] have very similar performance (around 84%), however when several models obtained using different values of C are used *together*, performance can be further increased (the “4 models” result in table 2). This suggests that the partitions obtained through different values of C capture different patterns in the class files, and that these patterns complement each other when used to classify unseen data.

Models obtained using the Bayesian Information Criterion have the lowest performance. We believe that the reason for this is the relatively larger number of states obtained per model using this approach, since a larger number of clusters represent a more detailed model which could overfit the training data. The models obtained by the GMM-EM approach with the low entropy criterion have the best performance.

Figure 2 shows the variation with discriminative training iteration of the classification performance. As in any gradient descent algorithm the initial search point has a crucial impact in the results obtained. The models found by the GMM-EM approach with the low occupancy criterion turned to be the best initial point for discriminative training. At their performance peak, the models reached a classification accuracy of over 90%, which is similar to the one obtained through SVMs in [3]. Analyzing the sources of improvement, we found that discriminative training effectively resolves the frequent confusion of oboe files with altotrombone files and the confusion between cellobowed files and violinbowed files. The acoustic differences between these two pairs are so small that they can even confuse a human classifier. Classification accuracy start to decrease after a certain number of iterations due to overfitting. A cross-validation set is needed to prevent the overfitting of the data during discriminative training, unavailable in this case due to the limited size of the database.

The right side of figure 2 shows classification performance in the case where every model is constrained to have the same number of states, as a function of the model size. A maximum performance of 81.8% is obtained for models with 5 states, but this is inferior to every other method of state number selection except BIC.

6. CONCLUSIONS AND FUTURE WORK

Hidden Markov Models are powerful tools for modeling general audio, finding spectral and temporal patterns in audio files with similar features, and generalizing their characteristics to form “basic” units of sound that can be used to decode streams of audio on the fly. However, performance depends critically on the initial HMM parameters, particularly the number of states and the transition matrix (topology constraints). We have presented a set of techniques that assign resources (states) to “class” models depending of the complexity and diversity of the sounds within the given class. Discriminative training permits us to resolve the uncertainty occasioned by similar competing classes, although a cross-validation dataset is required to control possible overfitting of the data.

This is our first work on generic audio processing, and we have used a paradigm in which a given segment of audio as a member of only one audio class. In future work, we will center our attention in modeling audio segments as mixtures of different class models.

7. REFERENCES

[1] M, Casey, “Reduced-Rank Spectra and Minimum-Entropy Priors as Consistent and Reliable Cues for Generalized Sound Recognition ,” *Proc Eurospeech*, Aalborg, September 2001.

[2] P. Gaunard, C.G.Mubikangiey, C. Couvreur and V. Fontaine, “Automatic Classification of Environmental Noise Events by Hidden Markov Models,” *Proc. ICASSP*, Seattle, 1998.

[3] G. Guo and S. Z. Li, “Content-based audio classification and retrieval using SVM learning” *IEEE Pacific-Rim Conference on Multimedia, Invited Talk* Australia, 2000.

[4] C. Rathinavelu, and L. Deng, “The Trended HMM with Discriminative Training for Phonetic Classification,” *ICSLP*, 1996.

[5] M.J. Reyes-Gomez and D.P.W. Ellis. “Error visualization for tandem acoustic modeling on the Aurora task.” *ICASSP*, Orlando, May 2002

[6] G. Schwarz “Estimating the dimension of a model,” *Ann. Statistic* Vol. 6, no. 2 1978

[7] E. Wold, T. Blum, D. Keislar and J. Wheaton, “Content-Based Classification, Search, and Retrieval of Audio,” *IEEE Multimedia*, 1996.

[8] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev and P. Woodland, “The HTKBook” 2000.

[9] T. Zhang and J. Kuo, “Content-Based Classification and Retrieval of Audio,” *Proc. ICASSP*, 1999.