USING THE FISHER KERNEL METHOD FOR WEB AUDIO CLASSIFICATION

*Pedro J. Moreno and Ryan Rifkin**

Compaq Computer Corporation
Cambridge Research Laboratory
One Kendall Square, Building 700
Cambridge, Massachusetts 02139
United States

ABSTRACT

As the multimedia content of the Web increases techniques to automatically classify this content become more important. In this paper we present a system to classify audio files collected from the Web. The system classifies any audio file as belonging to one of three categories: 'speech', 'music' and 'other'.

To classify the audio files, we use the technique of Fisher kernels. The technique as proposed by Jaakkola assumes a probabilistic generative model for the data, in our case a Gaussian mixture model. Then a discriminative classifier uses the GMM as an intermediate step to produce appropriate feature vectors. Support Vector Machines are our choice of discriminative classifier.

We present classification results on a collection of more than 173 hours of Web audio randomly collected. We believe our results represent one of the first realistic studies of audio classification performance on "found" data. Our final system yielded a classification rate of 81.8%.

1. INTRODUCTION

As the magnitude and use of multimedia content on the web grows, efficient ways to automatically find the "gist" of the contents become necessary. For example a search of all "audio" objects on altavista.com reports more than 600,000 different audio files (as of October 1999). Clearly, adding tags to help classify this content is necessary.

Previous work in the area of multimedia classification has focused mostly on using the surrounding text to label the multimedia object. For example, the current Altavista image/video/audio search technology uses only the surrounding text. Newer approaches combine the textual information surrounding the image with the image content itself [1]. With respect to audio classification, most previous systems have focused on very small databases and have used simple likelihood ratios of Gaussian distributions for classification. For example, [2] base their experiments on a database of 400 audio files selected by hand from clean recordings. It is clear that more sophisticated techniques are needed when the quality of the audio recordings degrades and the size of the database grows. The Fisher Kernel method is one such technique.

In this paper we investigate the problem of automatically classifying large collections of audio files into three broad categories: speech, music and other. The audio files are found on the web, with no control over quality or recording conditions. Such a simple classification can be thought of as a preliminary step in a more detailed labeling of audio files. For example, music files can be further classified by the music style, speech files can be transcribed into text with an automatic speech recognition engine and so on.

The outline of the paper is as follows. In Section 2 we give an overview of the Fisher kernel classification method. In Section 3 we describe the database used for our experiments as well as the choice of feature vectors. In Section 4 we present our experimental results. Finally, in Section 5 we present out conclusions and suggestions for future work.

2. OVERVIEW OF FISHER KERNEL CLASSIFICATION METHOD

The statistical modeling approach to a sequence of audio (speech, music, etc) feature vectors involves constructing a *generative* probability model, such as an HMM or a mixture of Gaussians. See for example [3]. Audio Sequences known to belong to a particular class are used as *training examples* to estimate the parameters of the generative model. Each class model assigns a probability that the audio sequence has been generated by it.

A generative method focuses on *explaining* the training data. A discriminative model, on the other hand, focuses on *finding the boundary* between classes in some feature space. Because of this property, discriminative methods often outperform generative models at classification. A major difficulty with using a discriminative methods for audio classification is that each audio file X consists of a sequence $\{x_1, \ldots, x_n\}$, where n varies among audio files; discriminative methods require a *fixed length* feature vector. The Fisher kernel approach allows us to overcome this difficulty.

* Ryan Rifkin is a PhD student at the Center for Biological and Computational Learning, AI Lab, Massachusetts Institute of Technology, MA, USA. This work was performed during a summer internship at Cambridge Research Laboratories, MA.

2.1. Generative model description

We define the probability density function for class c (out of a total of C classes) as a mixture of K Gaussians:

$$p(x_i|Class = c) = \sum_{l=1}^{K} P(l)p(x_i, \mu_{l,c}, \Sigma_{l,c}) \quad (1)$$

where x_i is a feature vector of dimension D, and $p(x_i, \mu_{l,c}, \Sigma_{l,c})$ is a multivariate Gaussian distribution with mean $\mu_{l,c}$ and covariance matrix $\Sigma_{l,c}$. In this paper, we assume a diagonal covariance matrix.

We further assume that each audio class has a particular known *a priori* probability $P(Class = c)$. If $X = \{x_1, x_2, \ldots, x_n\}$ denotes a single audio feature vector sequence with n vectors, and assuming that each vector in the sequence is independent and identically distributed, then the likelihood that the whole sequence has been generated by class c is $P(X|Class = c) = \prod_{i=1}^{N} p(x_i|Class = c)$. To decide which class is most likely we can apply Bayes rule and obtain

$$P(Class = c|X) = \frac{P(X|c)P(c)}{\sum_{i=1}^{C} P(X|i)P(i)} \quad (2)$$

If the generative model is itself to be used for classification, the class with the largest *a posteriori* probability is taken as an indication that the audio sequence has been generated by that class.

2.2. The Fisher Kernel

Instead of classifying directly with the generative model, we use the generative model to map audio sequences of variable length into a linear space of fixed dimension where discriminative techniques are applicable. Given a set of labeled audio sequences $\{X_1, \ldots, X_n\}$, each composed of a different number of feature vectors, we would like to find a discriminative classifier that separates sequences optimally.

We define a new feature vector, the Fisher score, as

$$U_X = \nabla_\theta log(P(X|\theta)) \quad (3)$$

Each component of U_X is a derivative of the log-likelihood of the audio sequence X with respect to a particular parameter of the generative model. In our case the parameters θ of the generative model are the prior probability for each mixture Gaussian $P(l)$, each component of the mean vector $\mu_{l,c}$ and the diagonal covariance matrix $\Sigma_{l,c}$.

Once a generative model is trained, for every training sequence of feature vectors $X_i = x_1, x_2, \ldots, x_{n_i}$, composed of n_i feature vectors we transform it into a vector of fixed dimension. For example, if we use the mean vectors as our model parameters θ, *i.e.*, for $\theta = \mu_l$ then the Fisher score is

$$\nabla_{\mu_l} log(P(X|\mu_l)) = \sum_{t=1}^{n_i} P(l|x_t)\Sigma_l^{-1}(x_t - \mu_l) \quad (4)$$

where $P(l|x_t)$ represents the *a posteriori* probability of mixture l given the observed feature vector x_t.

Similarly, by taking the derivatives we can obtain a Fisher score with respect to the prior probabilities $P(l)$ or with respect to the covariance matrices Σ_l. For more details on the general theory of Fisher kernel methods and for details on how to compute these Fisher scores see [4].

Once each original audio file sequence X_i with a variable number of feature vectors n_i has been transformed into a fixed dimensional Fisher score feature vector U_{X_i} we train/test our discriminative classifier. In our experiments we have used a Support Vector Machine (SVM) as classifier. Burges [5] provides a good introduction to the general theory of SVMs.

3. DATABASE DESCRIPTION AND PARAMETRIZATION

Through a random walk of the World Wide Web, we obtained a random list of 1,000,000 web pages with multimedia content. From this list we extracted those pages with audio content. We collected a random list of multimedia files in the following formats: RealNetworks, Microsoft MediaPlayer, Microsoft wave format, AVI videoformat and Apple QuickTime. For those multimedia files that contained audio and video we only retained the audio stream. All Audio files were resampled to a uniform sampling rate of 16,000 kHz with a single audio channel.

The final database contained 13,016 audio files. Its duration ranges from a minimum of half a second to a maximum of 7 minutes. The mean duration was 48 seconds. Figure 1 shows a histogram of file duration in seconds for our database.

We used the mel cepstra feature vector representation commonly used in speech recognition systems. We converted the audio files from their waveform representation into a sequence of 13 dimensional mel cepstral feature vectors with their time derivatives. The cepstra and its time derivatives were combined into a 26 dimensional vector. For our cepstral analysis we used a 25.5 ms Hamming window shifted every 10 ms. We evaluated the use of the second derivatives of the cepstrum vector but observed no significant gains. We also applied the cepstral mean normalization procedure typically applied in speech recognition systems.

Each audio file was labeled by humans and a detailed category was assigned to it. The categories describe music style, type of background noise, quality of the audio recording etc. In those cases in which the file had several sections with different audio categories, e.g., the first section was music, the second one speech with background music, the third section speech, etc, the labelers were instructed to pick the label that dominated. In cases in which no label was clearly dominating they were instructed to classify the audio file as "other".

Finally, all the labels were merged into three: speech, music and other. Table 1 presents some statistics of the resulting database.

4. EXPERIMENTS AND RESULTS

4.1. Gaussian Mixture Classifier

Using the well known EM algorithm [6] we trained three Gaussian mixture models for each of the classes, for use as generative models.

Class Label	Number of files	Duration hours:min:sec
speech	7121	101:40:00
music	4282	60:37:00
other	1631	11:21:00

Table 1: Duration and Population statistics for the Web Audio database.

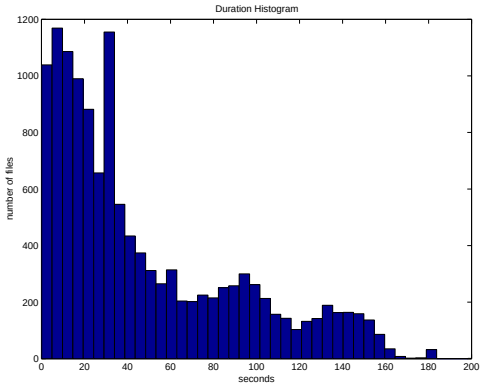

Figure 1: Histogram of file duration in seconds for the experimental database.

To obtain better estimates of the error rate for each of the classifier configurations we performed 10-fold cross validations: we split our database into ten equal subsets, and trained separately for each of the ten splits of these sets into nine parts traning and one part testing data. This allowed us to evaluate the mean and variance of the correct classification rate over the 10 non overlapping train/test partitions.

The only parameter configuration we explored was the number of Gaussian mixtures. Table 2 presents our results.

Number of Gaussians	Correct Rate mean	var
20	0.623	0.020
68	0.684	0.012
100	0.674	0.011
116	0.673	0.012
132	0.670	0.011

Table 2: Gaussian Mixture classification performance vs. number of mixtures.

The best classification rate was obtained with a system with 68 mixtures per component. A further increase in the number of mixtures did not yield any improvements. For our experiments with Fisher kernel methods we used this configuration as the intermediate generative model.

4.2. Fisher Kernel based SVM Classifier

We partitioned the database into 3 sections. The first section contained 50% of the corpora and was used for training the Gaussian mixture classifiers. The 3 sets of Gaussian mixtures (68 components per classifier) were combined into a single one, reweighting the prior probabilities to make sure they added up to 1.0. Given the parameters θ of the generative model chosen to take derivatives from, the remaining 50% of the corpora was transformed into Fisher score vectors. This new corpora was split into training and testing sets. The training set contained 40% of the whole corpora while the testing set contained the remaining 10% of the corpora. This partition of the whole database was performed ten times so we could obtain estimates of the error vars for the classification accuracy.

For our SMVs, we decided to explore the choice of kernel and capacity (C). We experimented with polynomial kernels of orders two and three and with Gaussian kernels. None of the polynomial kernels yielded good results; all results reported in the paper are based on Gaussian kernels. In general, as we increase the capacity, we work harder to enforce complete separation *on the training set*; this may or may not lead to improved generalization error.

4.2.1. One vs. *One and One* vs. *All Classifiers*

Since our database contains more than two classes and SMVs are inherently two class classifiers, we need to combine individual classifiers to obtain a global classification.

Two common schemes for combining SVM classifiers are one *vs* all and one *vs* one. In one *vs* all, n classifiers are trained, each of which attempts to distinguish a single class from all remaining classes. A data point is classified as belonging to a class I if the I *vs* all classifier gives a YES vote and all the other classifiers give NO votes. In the one *vs* one scheme, $n(n-1)/2$ classifiers are trained, and a data point is classified as the class with the most YES votes.

In our version of the scheme, if a point is not classified unambiguously, it is assigned to one of the three classes completely at random.

4.3. Results

To generate Fisher score feature vectors θ we explored the use of prior probabilities $P(l)$, mean vectors μ_l, and the covariance matrix Σ_l. The covariance matrix Fisher score results were poor and hence are not reported here. Table 3 presents classification results (on the testing set) for different values of the SVM capacity, using a one *vs* one combination scheme.

As we can see in the table a Fisher score based on mean vectors yields slightly better results. However, the resulting feature vector contains 5304 components (three sets of 68 Gaussian mixtures multiplied by 26 components per mean vector) compared with the 204 components in the prior based feature vector (three sets of 68 priors).

Table 4 compares the classification performance of the one *vs* one and the one *vs* all multiclass schemes for multiple choices of the SVM capacity, using the Fisher score based on mean vectors.

As we can see, the one *vs* one multiclass scheme provided better classification results. Further more, the one *vs* one scheme was less sensitive to the capacity value. Similar results were observed for the $\theta = P(l)$ feature vector.

Capacity	$\theta = P(l)$ mean(var)	$\theta = \mu_l$ mean(var)
1	0.765 (0.015)	0.774 (0.013)
10	0.785 (0.014)	0.799 (0.012)
100	0.795 (0.012)	0.797 (0.012)
1000	0.784 (0.012)	0.764 (0.007)
10000	0.767 (0.013)	0.749 (0.009)

Table 3: Classification performance (mean and variance) *vs* SVM capacity using a Gaussian kernel and a SVM one *vs* one multiclass architecture.

Capacity	one *vs* one mean(var)	one *vs* all mean(var)
1	0.774 (0.013)	0.773 (0.007)
10	0.799 (0.012)	0.797 (0.005)
100	0.797 (0.012)	0.781 (0.009)
1000	0.764 (0.007)	0.730 (0.009)
10000	0.749 (0.009)	0.690 (0.005)

Table 4: Classification performance (mean and variance) *vs* SVM capacity for two SVM multiclass architectures. All results based on a $\theta = \mu_l$ Fisher score vector and a Gaussian kernel based SVMs.

4.3.1. A More Complex Multiclass Scheme

In our application, there are only three classes, and the one *vs* one and one *vs* all scheme each require only three classes. Furthermore, a approximately 10% of the points were ambiguously classified by the one *vs* all scheme (all three classifiers returned a negative vote or two or more classifiers returned a positive vote). Therefore, we experimented in combining both schemes. We first classified the testing data using the one *vs* all scheme. For those cases in which the results where ambiguous , the data was sent to the one *vs* one classifier for final disambiguation. All experiments were performed with the choice of $\theta = \mu_l$ and Gaussian kernels for all SMVs. Our best classification result was obtained with a capacity of 100.0 and returned a classification rate of 0.8179 with a variance of 0.0457.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have explored the use of Fisher kernel methods for large scale web audio classification. The Fisher kernel method represents a novel technique that combines the benefits of generative models and discriminative classifiers. It takes advantage of the expressive power of generative models to map sequences of features of variable length, such as audio sequences, into a fixed length representation. This is fundamental in allowing the use of discriminative classifiers such as SMVs.

We believe this is one of the first results reported on a large audio corpora (173 hours) *randomly*[1] collected. While other authors have reported better classification performance on related tasks, their results are not directly comparable to ours. In some cases their classification results have been based on short homogeneous audio segments [3]. In others their results have been based on small databases carefully selected and more geared to audio similarity and retrieval [7].

For future research we will explore how to produce a more detailed labeling structure for the audio. Since the database contains fine grained labels indicating music style, noise level etc, it should be possible to build classifiers that try to guess these labels. We will also study the classification and retrieval performance of the system when the text surrounding the audio is also used for classification.

6. ACKNOWLEDGMENTS

We would like to thank Gene Preble and John Axon for their work in labeling the audio content. We also would like to thank Janet Marques for her help in designing the labeling software. Also R. Paul Johnson and Chris Weikart for their help in downloading the audio content and in the use of the web audio crawler. We also thank altavista.com engineering for providing the list of multimedia url's and allowing us access to their crawler logs.

7. REFERENCES

[1] S. S. M. La Cascia and S. Sclaroff, "Combining textual and visual cues for content-based image retrieval on the world wide web," in *Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries*, 1998.

[2] D. K. E. Wold, T. Blum and J. Wheaton, "Content-based classification, search, and retrieval of audio," *IEEE Transactions on Multimedia*, vol. 3, Sept. 1996.

[3] M. A. Siegler, U. Jain, B. Raj, and R. M. Stern, "Automatic segmentation, classification and clustering of broadcast news data," in *DARPA Speech Recognition Workshop*, pp. 97–99, 1997.

[4] M. D. T. Jaakkola and D. Haussler, "A discriminative framework for detecting remote protein homologies," *Journal of Computational Biology*, 1998.

[5] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, 1998.

[6] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data Using the EM Algorithm," *Journal of the Royal Society of Statistics*, vol. 39, no. 1, pp. 1–38, 1977.

[7] J. T. Foote, "Content-based retrieval of music and audio," in *SPIE*, pp. 138–147, 1997.

[1] No effort whatsoever was made to filter or clean the database in any way. The database presents an accurate view of the Web at the time of its collection.