Content-Based Audio Classification and Retrieval Using SVM Learning

Stan Z. Li* GuoDong Guo
Microsoft Research China, 5/F Beijing Sigma Center, Beijing 100080, China
* szli@microsoft.com

Abstract

In this paper, a support vector machines (SVMs) based method is proposed for content-based audio classification and retrieval. Given a feature set, which in this work is composed of perceptual and cepstral feature, optimal class boundaries between classes are learned from training data by using SVMs. Matches are ranked by using distances from boundaries. Experiments are presented to compare various classification methods and feature sets.

1 Introduction

Audio data is an integral part of many modern computer and multimedia applications. Numerous audio recordings are dealt with in audio and multimedia applications. The effectiveness of their deployment is greatly dependent on the ability to classify and retrieve the audio files in terms of their sound properties or content. Rapid increase in the amount of audio data demands for a computerized method which allows efficient and automated content-based classification and retrieval of audio database. For these reasons, commercial companies developing audio retrieval products are emerging.

Wold *et al.* [14] have developed a system called "Muscle Fish". That work distinguishes itself from earlier work [5, 3, 2] in its content-based capability. There, various perceptual are used to represent a sound. A normalized Euclidean (Mahalanobis) distance and the *nearest neighbor* (NN) rule are used to classify the query sound into one of the sound classes in the database. In Liu *et al.* [10], separability of different classes is evaluated in terms of the intra- and inter-class scatters to identify highly correlated features. Foote [4] choose to use 12 mel-frequency cepstral coefficients (MFCCs) as the audio features. Histograms of sounds are compared and the classification is done by using the NN rule. In Pfeiffer *et al.* [12], audio features are extracted by using gammaphone filters.

Recently, a new pattern recognition method, called the *Nearest Feature Line* (NFL), is developed. This method explores information contained in multiple prototypes per class by using linear interpolation and extrapolation of each pair of prototypes in the class. It has been shown to produce better results than Euclidean distance based ranking methods such as k-NN in face recognition [9], image [8] and audio [7] classification and retrieval.

In this paper, a support vector machines (SVMs) [1, 13] based method is proposed for content-based audio classification and retrieval. The SVM minimizes the structural risk, that is, the probability of misclassifying yet-to-be-seen patterns for a fixed but unknown probability distribution of the data. This is in contrast to traditional pattern recognition techniques of minimizing the empirical risk, that is, of to optimizing the performance on the training data. This minimum structural risk principle is equivalent to minimizing an upper bound on the generalization error.

Given a class-labeled training set, which in this work is a set of labeled feature vectors composed of perceptual and cepstral feature (Section 2), class boundaries between each pair of two classes are learned using SVMs (Section 3. A binary tree is formed for multi-class problems. A new metric called distance-from-boundary (DFB) is used to measure and rank similar audio patterns. Extensive experiments are performed (Section 3) to compare SVMs with NFL, NN and MuscleFish using a database of 409 sounds from MuscleFish.

2 Audio Feature Selection

Before feature extraction, an audio signal (8-bit ISDN μ-law encoding) is preemphasized with parameter 0.96 and then divided into frames. Given the sampling frequency of 8000 Hz, the frames are of 256 samples (32ms) each, with 25% (64 samples or 8ms) overlap in each of the two adjacent frames. A frame is hamming-windowed by $w_i = 0.54 - 0.46 * \cos(2\pi i/256)$. It is marked as a silent frame if $\sum_{i=1}^{256}(w_i s_i)^2 < 400^2$ where s_i is the preemphasized signal magnitude at i and 400^2 is an empirical threshold.

Two types of features are computed from each frame: (i) perceptual features, composed of total power, sub-band

powers, brightness, bandwidth and pitch; and (ii) mel-frequency cepstral coefficients (MFCCs). Then audio features are extracted from each non-silent frame. The means and standard deviations of the feature trajectories over all the non-silent frames are computed, and these statistics are considered as feature sets for the audio sound.

The means and standard deviations of the above 8 original perceptual features are computed over the non-silent frames, giving two feature vectors of 8-dimension each. The two vectors are concatenated to form a 16-dimensional vector. Adding the silence ratio (number of silent frames/total number of frames) and the pitched ratio (number of pitched frames/total number of frames) to this vector gives an augmented 18-dimensional perceptual feature vector, named "perc". Each x_i of the 18 components in the perc set is normalized according to $x_i' = (x_i - \mu_i)/\sigma_i$ (correlations between different features are ignored) where the mean μ_i and standard deviation σ_i are calculated over all the training set. This gives the final perceptual feature set, named "Perc".

Note the following differences between the perceptual features used in this work and in Muscle Fish: First, the two sets of perceptual features are different. Second, in Muscle Fish, there is no concatenation of the original features and their standard deviations into an augmented vector; instead, the instantaneous derivative (time differences) for all of their perceptual features are used as additional features. Third, in Muscle Fish, the normalization is carried out in the calculation of the Mahalanobis distance by using the means and covariance matrix.

The means and standard deviations of the L MFCCs are calculated over the non-silent frames, giving a $2L$-dimensional cepstral feature vector, named "Ceps$L$". In the experiments, Ceps$L$ with L values in the range between 5 and 120, with the corresponding feature sets named Ceps5, $\cdots$, Ceps120, are evaluated.

Note that the cepstral coefficients are not normalized. Empirically, the normalization of the perc set into Perc set by the mean and standard deviation gives better results whereas a similar normalization of the cepstral vectors leads to worse results.

The Perc and Ceps$L$ feature sets are weighted and then concatenated into still another feature set, named "PercCeps$L$", of dimension $18 + 2L$ (see [7] for more details), giving PercCeps5, $\cdots$, PercCeps120. See [7] for detailed definitions of these features.

3 Learning Using Support Vector Machines

3.1 Linear Support Vector Machines

Consider the problem of separating the set of training vectors belonging to two separate classes, $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)$, where $\mathbf{x}_i \in R^n$ is a feature vector and $y_i \in \{-1, +1\}$ a class label, with a hyperplane of equation $\mathbf{w}\mathbf{x} + b = 0$. Of all the boundaries determined by $\mathbf{w}$ and b, the one that maximizes the margin (Fig.1.Left) would generalize well as opposed to other possible separating hyperplanes.

A canonical hyperplane [13] has the constraint for parameters $\mathbf{w}$ and b: $\min_{\mathbf{x}_i} y_i(\mathbf{w} \cdot \mathbf{x}_i + b) = 1$. A separating hyperplane in canonical form must satisfy the following constraints, $y_i\left[(\mathbf{w} \cdot \mathbf{x}_i) + b\right] \geq 1, \;\; i = 1, \ldots, l$. The margin is $\frac{2}{\|\mathbf{w}\|}$ according to its definition. Hence the hyperplane that optimally separates the data is the one that minimizes $\Phi(\mathbf{w}) = \frac{1}{2} \parallel \mathbf{w} \parallel^2$.

The solution to the optimization problem can be obtained as follows [13]: First, find the maximization solution to the following problem

$$\bar{\alpha} = \arg\max_{\alpha} \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \qquad (1)$$

subject to

$$\alpha_i \geq 0 \; (i = 1, \ldots, l), \quad \sum_{i=1}^{l} \alpha_i y_i = 0 \qquad (2)$$

Then claculate

$$\bar{\mathbf{w}} = \sum_{i=1}^{l} \bar{\alpha}_i y_i \mathbf{x}_i, \quad \bar{b} = -\frac{1}{2} \bar{\mathbf{w}} \cdot [\mathbf{x}_+ + \mathbf{x}_-] \qquad (3)$$

where $\mathbf{x}_+$ is a support vector of the "+" class and $\mathbf{x}_-$ is a support vector of the "-" class. Now, a new data point $\mathbf{x}$is classified by the sign of

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{l} \bar{\alpha}_i y_i (\mathbf{x}_i \cdot \mathbf{x}) + \bar{b}\right) \qquad (4)$$

3.2 Linearly Non-Separable Case

In non-separable cases, slack variables $\xi_i \geq 0$, which measure the mis-classification errors, can be introduced and

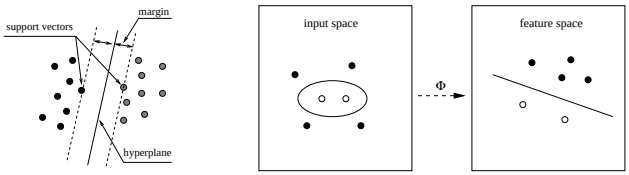

Figure 1. (Left) A linear SVM. (Right) A non-linear SVM.

a penalty function, $F(\xi) = \sum_{i=1}^{l} \xi_i$, added to the objective function [1]. The optimization problem is now treated as to minimize the total classification error as well as the bound on the VC dimension [13] of the classifier. The solution is identical to the separable case except for a modification of the Lagrange multipliers as $0 \leq \alpha_i \leq C, \ \ i = 1, \ldots, l$. We refer to [13] for more details on the non-separable case.

3.3 Kernel Support Vector Machines

In linearly non-separable but nonlinearly (better) separable case, the SVM replaces the inner product $\mathbf{x} \cdot \mathbf{y}$ by a kernel function $K(\mathbf{x}, \mathbf{y})$, and then constructs an optimal separating hyperplane in the mapped space. According to the Mercer theorem [13], the kernel function implicitly maps the input vectors, via a $\mathbf{\Phi}$ associated with the kernel, into a high dimensional feature space in which the mapped data is linearly separable (Fig.1.Right). This provides a way to address the curse of dimensionality [13]. Possible choices of kernel functions include (1) Polynomial $K(\mathbf{x}, \mathbf{y}) = ((\mathbf{x} \cdot \mathbf{y} + 1))^d$, where the parameter d is the degree of the polynomial; (2) Gaussian Radial Basis Function $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{(\mathbf{x}-\mathbf{y})^2}{2\sigma^2}\right)$, where the parameter σ is the width of the Gaussian function; and (3) Multi-Layer Perception $K(\mathbf{x}, \mathbf{y}) = \tanh\left(\kappa(\mathbf{x} \cdot \mathbf{y}) - \mu\right)$, where the the κ and μ are the scale and offset parameters. However, Exponential Radial Basis Function (ERBF) $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{|\mathbf{x}-\mathbf{y}|}{2\sigma^2}\right)$ has been empirically observed to perform better than above three [6]. For a given kernel function, the classifier is given by $f(\mathbf{x}) = sign\left(\sum_{i=1}^{l} \bar{\alpha}_i y_i K(\mathbf{x}_i, \mathbf{x}) + \bar{b}\right)$.

3.4 Multi-Classes

Classification of multi-classes can be achieved by combining all the two-class SVMs. There are two common schemes for this purpose: one-against-all and the one-against-one. We use the latter and construct a bottom-up binary tree for classification. By comparison between each pair, one class number is chosen representing the "winner" of the current two classes. The selected classes (from the lowest level of the binary tree) will come to the upper level for another round of tests. Finally, a unique class label will appear on the top of the tree.

Denote the number of classes as c, the SVMs learn $\frac{c(c-1)}{2}$ discrimination functions in the training stage, and carry out comparisons of $c-1$ times under the fixed binary tree structure. If c is not equal to a power of 2, we can decompose c as: $c = 2^{n_1} + 2^{n_2} + \ldots + 2^{n_I}$, where $n_1 \geq n_2 \geq \ldots \geq n_I$, because any natural number (even or odd) can be decomposed into finite positive integers which are the power of 2.

3.5 Metric for Ranking

Given a query $\mathbf{q}$, the classification and retrieval is ranked by using the signed distances from $\mathbf{q}$ to the c boundaries as follows. The k^{th} boundary separating class k from the others is defined by $(\mathbf{x}_k^*, \beta_k^*, b_k^*)$, where $\mathbf{x}_k^* = bfx_{kj}^* \mid j = 1, \cdots, m_k\}$ are the m_k support vectors, $\beta_k^* = \{\beta_{kj}^* = \alpha_{kj}^* y_{kj}^*\}$ and $b_k^* = \{b_{kj}^*\}$ the optimal coefficients. The signed distance of $\mathbf{q}$ to the k^{th} boundary is calculated by

$$D(\mathbf{q}; \beta_k^*, \mathbf{x}_k^*, b_k^*) = \frac{\sum_{j=1}^{k_m} \beta_{kj}^* K\left(\mathbf{x}_{kj}^*, \mathbf{q}\right) + b_k^*}{\| \sum_{j=1}^{k_m} \beta_{kj}^* \mathbf{x}_{kj}^* \|} \qquad (5)$$

The patterns within the same class have positive distances to their enclosing boundary, while other patterns have negative distance to this boundary. The top matched class is found by $k^* = \arg\max_{1 \leq k \leq c} D(\mathbf{q}; \beta_k^*, \mathbf{x}_k^*, b_k^*)$

4 Experiments

The experiments compare the SVM (using ERBF kernel with $\sigma = 4$ and $C = 200$) with the NFL method using the NN as the baseline, and also with the MuscleFish system, for the Perc, Ceps, PercCeps feature sets. An audio database of 409 sounds from MuscleFish is used for the experiments, which is manually classified into 16 classes [14]. The data of each class c containing N_c sounds is randomly partitioned into a prototype (training) set and a test set, with $ceiling(N_c/2)$ sounds in the training set and $floor(N_c/2)$ in the test set, resulting in total numbers of 211 sounds in the former and 198 in the latter altogether. 10 such random partitions are obtained.

The *error rate* is used to measure the classification performance, which is averaged over all the queries in a test set and over all the 10 random partitions. The average *retrieval accuracy* [11] is calculated similarly as the retrieval performance measure. Among the results for Ceps5, Ceps10, $\cdots$, Ceps120, only that for Ceps40 is shown because the SVM achieved best results with that. Among the results for PercCeps$N$, only PercCeps8 is shown for the same reason.

Table 1 shows the average error rates of SVM (linear and kernel-based), NFL and NN obtained by using the separate training and test sets, for the three types of feature sets, in which the absences of some l-SVM data are due to its non-convergence. Fig.2 shows the average retrieval efficiencies, with the results obtained by using the leave-one-out (LOO) test also shown.

From the results, we see that both SVM and NFL are better than NN in terms both of the error rate and retrieval efficiency. SVM has error rates similar to NFL but the lowest error rate is obtained by using NFL with the PercCeps8

Feature Set	l-SVM	k-SVM	NFL	NN
Perc	18.74	**15.05**	17.12	19.80
Ceps40	24.75	**22.22**	22.42	31.47
PercCeps8	17.93	14.09	**13.18**	18.54

Table 1. Average error rates (%).

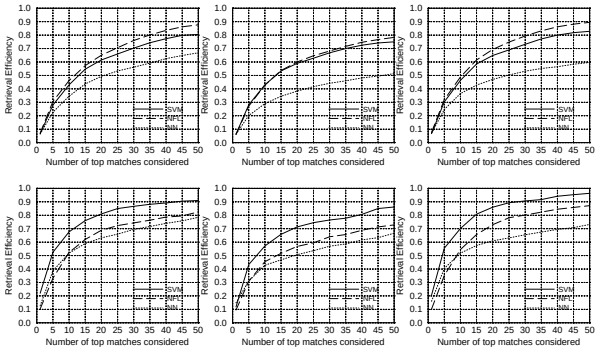

Figure 2. Average retrieval efficiencies obtained by using LOO test on a single database (top) and by using separate training and test sets (bottom).

feature. SVM has a bit lower retrieval efficiencies than NFL by the LOO test but higher by using the separate training and test sets.

Complexity wise, SVM takes long time to train whereas NFL needs no training. During classification, NFL's complexity is linear to the number of classes C whereas SVM's is proportional to $C * (C-1)/2$ when the binary tree recognition strategy is used; within a class c, SVM's is proportional the number of supporting vectors whereas NFL's is proportional to $N_c * (N_c - 1)/2$ where N_c is the number of prototypes in c. Note that the nearest class center (NC), and the k-NN (for $k > 1$), which also make use of class information, produce less favorable results than the simple 1-NN in all the applications we have evaluated [9, 7, 8].

Finally, the results are compared with that of MuscleFish [14] in terms of the error rates obtained by LOO test. The MuscleFish method is equivalent to a perceptual feature set with an NN rule. Its error rate, which is obtained from `http://www.musclefish.com/cbrdemo.html`, is 19.07% (78 errors out of 409 queries). In comparison, the error rates of the NN+Perc method is 13.94% (57 errors), which means our Perc feature set is better than MuscleFish's. Lower error rates are obtained using the PercCeps8 feature set: 11.00% (45 errors) by kernel SVM and 9.78% (40 errors) by NFL.

5 Conclusion

A SVM based method is proposed for content-based audio classification and retrieval. Like NFL, the SVM has good performance in audio classification and retrieval, better than currently achieved by the MuscleFish system. When tested using separate training and test sets, SVM is more advantageous than NFL in terms of retrieval efficiency, demonstrating its said generalization ability to classify patterns unseen in the training set. However, SVM takes long time to train, and currently, and needs to select kernel function and parameter therein which is currently practiced by trial and error.

//

References

[1] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[2] B. Feiten and S. Günzel. "Automatic indexing of a sound database using self-organizing neural nets". *Computer Music Journal*, 18(3):53–65, 1994.

[3] B. Feiten and T. Ungvary. "Organizing sounds with neural nets". In *Proceedings 1991 International Computer Music Conference*, San Francisco, 1991.

[4] J. Foote. "Content-based retrieval of music and audio". In C. C. J. Kuo et al., editors, *Multimedia Storage and Archiving Systems II, Proc. of SPIE*, volume 3229, pages 138–147, 1997.

[5] S. Foster, W. Schloss, and A. J. Rockmore. "Towards an intelligent editor of digital audio: Signal processing methods". *Computer Music Jounal*, 6(1):42–51, 1982.

[6] S. Gunn. "Support vector machines for classification and regression". Technical report, ISIS, May 1998.

[7] S. Z. Li. "Content-based classification and retrieval of audio using the nearest feature line method". *IEEE Transactions on Speech and Audio Processing*, September 2000.

[8] S. Z. Li, K. L. Chan, and C. L. Wang. "The the nearest feature line method in image classification and retrieval". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, -(-):–, accepted 2000.

[9] S. Z. Li and J. Lu. "Face recognition using the nearest feature line method". *IEEE Transactions on Neural Networks*, 10(2):439–443, March 1999.

[10] Z. Liu, J. Huang, Y. Wang, and T. Chen. "Audio feature extraction and analysis for scene classification". In *IEEE Signal Processing Society 1997 Workshop on Multimedia Signal Processing*, 1997. http://vision.poly.edu:8080/paper/audio-mmsp.html.

[11] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, August 1996.

[12] S. Pfeiffer, S. Fischer, and e. W. E. "Automatic audio content analysis". Tech. Rep. No. 96-008, University of Mannheim, Mannheim, Germany, April 1996. ftp://pi4.informatik.uni-mannheim.de/pub/techreports/1996/TR-96-008.ps.gz.

[13] V. N. Vapnik. *Statistical learning theory*. John Wiley & Sons, New York, 1998.

[14] E. Wold, T. Blum, D. Keislar, and J. Wheaton. "Content-based classification, search and retrieval of audio". *IEEE Multimedia Magazine*, 3(3):27–36, 1996. http://musclefish.musclefish.com/ieeemm96/.