Fast SVM Training based on the Choice of Effective Samples for Audio Classification

Shilei Zhang, Hongchen Jiang, Shuwu Zhang, Bo Xu

Institute of Automation, Chinese Academy of Sciences, Beijing, 100080, China
{slzhang, hcjiang, swzhang, xubo}@hitic.ia.ac.cn

Abstract

In this paper, we propose a new method to choose the effective samples for support vector machines (SVM) training based on regression tree in audio classification task. The objective is to reduce the training time of SVM by choosing effective examples from the training set and to balance the number of training points of binary classes. One obvious advantage of such method is that it provides a flexible framework to implement the choice procedure based on the training data for a given classification task. We test the performances of our new method on a dataset composed of about 6-hour audio data which illustrate that the computation time can be significantly reduced without a significant decrease in the prediction accuracy.

Index Terms: SVM, regression tree, effective examples, audio classification

1. Introduction

Audio signals which include speech, music and environmental noise are important types of media. With the rapid increase of multimedia information, the problem of distinguishing audio signals into these different audio types is thus becoming increasingly significant. Content-based audio classification and segmentation is broadly used in speech recognition, audio archive management, audio searching and indexing etc. Various methods for audio discrimination have been proposed for the needs of different applications [1, 2].

SVM is a very effective classifier algorithm for audio classification. Support vector machines are derived from the idea of the generalized optimal hyperplane with maximum margin between the two classes and this idea implements the structural risk minimization principle in the statistical learning theory. Maximizing the margin plays an important role in the capacity control so that the SVM will not only have small empirical risks, but also have good generalization performance. Various training algorithms have been proposed to speed up the training, including chunking, decomposition method, and Platt's Sequential Minimal Optimization (SMO) [3]. Although these algorithms have been proven to accelerate the training, they do not scale well with the size of the training data.

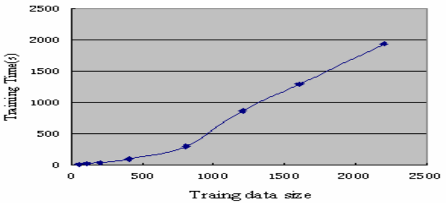

Figure1: *Training time versus the size of training set.*

Fig. 1 shows the training time with the growing size of training set using the LIBSVM package [4]. We can learn about standard SVM training has O(m^3) time and O(m^2) space complexities, where m is the training set size. Thus, when handling a large amount of data in machine learning, it is important to reduce the computation complexity and memory requirement without degrading the prediction accuracy. Research in this field has gained a lot of attention in these past few years. Various methods to reduce the size of the training dataset are proposed, including Vector Quantization (VQ) [5], probabilistic estimates related to editing algorithms [6] and clustering [7, 8]. In this paper, we present a new tree-based effective training samples selection method. In other words, regression tree is built through a process known as binary recursive partitioning where each decision node in the tree contains a training subset from the whole dataset; then we choose and balance the most qualified and effective samples for binary classes via data-driven algorithm.

The rest of the paper is organized as follows: In section 2 we introduce the concepts of SVM and LIBSVM software package. The framework of audio classification will be discussed in section 3. Section 4 describes the proposed training samples choice procedure. In section 5 experiments are presented and the conclusions will be drawn in section 6.

2. Introduction to SVM

2.1. Basic theory of SVM

The SVM is a discriminative classifier that is simple in concept but has some extensions that make it very powerful. Here we focus on the C-SVM applied to the two-class pattern recognition problem. Given n training patterns x_i and their associated classes $y_i \in \{+1,-1\}$, the SVM decision function is:

$$f(x)=\sum_{i=1}^{n} y_i\alpha_i K(x_i,x)+b \qquad (1)$$

The coefficients α in (1) are obtained by solving a quadratic programming problem:

$$\max_{\alpha}\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j K(x_i,x_j) \qquad (2)$$

$$\text{subject to } \forall i, 0\le\alpha_i\le C \text{ and } \sum_{i=1}^{n}\alpha_i y_i=0$$

This optimization yields three categories of training samples depending on α . Within each category, the possible values of the margins $y_i f(x_i)$ are prescribed by Karush-Kuhn-Tucker optimality conditions [6].

- Samples corresponding to $\alpha_i = C$ are called boundary support vectors (SVs) and satisfy $y_i f(x_i) < 1$. The set of boundary SVs includes all training samples misclassified by the SVM;

- Samples corresponding to $0 < \alpha_i < C$ are called standard SVs and satisfy $y_i f(x_i) = 1$;
- Samples corresponding to $\alpha_i = 0$ satisfy $y_i f(x_i) > 1$. These examples play no role in the SVM decision function (1). Retraining after discarding these examples would still yield the same SVM decision function.

Training a SVM requires the solution of a very large quadratic programming (QP) optimization problem. SMO breaks this large QP problem into a series of smallest possible QP problems.

For some classification problems, numbers of data in different classes are unbalanced. The decision boundary tends to be determined to make more correct decision for the larger class in order to maximize total accuracy. For handling this unbalanced data, different penalty parameters are proposed to use in the SVM formulation (2) to control this balance between false positives and false negatives. This creates a design boundary which has different distances from labelled points on the two sides: the class with higher loss will be given a larger margin. The difficulty of this simple approach lies in the proper selection of these penalty parameters.

2.2. LIBSVM package

Our SVM implementation is based on the LIBSVM, a library for SVM classification and regression. LIBSVM adopts an SMO-type method for SVM training strategy of solving quadratic problems. Radial basis function (RBF) defined as (3) is used as kernel. Model selection in this class of SVM involves two hyper-parameters: the penalty parameter C and the kernel width γ. The γ in the RBF kernel controls the shape of the kernel and C controls the trade-offs between margin maximization and error minimization. Increasing C may decrease training error, but it can also lead to poor generalization. We perform a grid-search on C and γ using 5 fold cross-validation. Basically pairs of (C, γ) are tried and the one with the best cross-validation accuracy is picked.

$$K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2), \gamma > 0 \qquad (3)$$

3. Hierarchical classification and feature selection

In this paper, multi-class audio classification is considered as an important preprocessing step to speech recognition, which classifies audio clips into one of four classes: pure speech, non-pure speech, music, noise. Audio classification is made up of two main sections: a signal processing section and a classification section.

We first uniformly segment the audio signal into non-overlapping 1s long clips, then the clip is further divided into non-overlapping 25ms long frames, next various features are extracted from each clip to represent it. Feature selection can select the most relevant features to help understand the problems from different fields. Nineteen kinds of audio features are considered in this work, which are chosen due to their effectiveness in capturing the temporal and spectral structures of different audio classes. The detail description of these features can be found in the references [2, 9]. These features are: Zero-Crossing Rate (ZCR), High ZCR Ratio (HZCRR), Short-Time Energy (STE), Low STE Ratio (LSTER), Root Mean Square (RMS), Silence Frame Ratio (SFR), Sub-band Energy Distribution (SED), Spectrum Flux (SF), Spectral Centroid (SC), Spectral Spread (SS), Spectral Rolloff Frequency (SRF), Sub-band Periodicity (BP), Noise Frame Ratio (NFR), Linear Spectrum Pair (LSP), Linear Predictive Cepstral Coefficients (LPCC), MEL-frequency Cepstral Coefficients (MFCC), Normalized RMS Variance (NRMSV), Joint RMS/ZC Measure (JRZM), 4Hz modulation energy (4HME). But for different audio classes, the effectiveness and robustness of these features are not identical, therefore we select different subsets of available features for different classification spaces.

We adopt hierarchical classification structure for the SVMs in a multi-class pattern recognition task. Silence segments are first detected and removed using a simple energy-based algorithm. Then the non-silence clips are classified into speech and non-speech by the SVM1 classifier. Next, speech signals are classified into pure speech and non-pure speech by the SVM2 classifier, while non-speech signals are further classified into music and environment noise by the SVM3 classifier, respectively. Based on experiments and analyses, we construct three groups of feature sets for the three SVM classifiers respectively, as shown in Table 1. This Framework will be applied to our following testing experiments.

Table1: *Three groups of feature sets*

	Feature Sets
SVM1	HZCRR、ZCR、LSTER、RMS、SC、SS、BP、NFR、SF、LPCC、LSP、MFCC
SVM2	SFR、ZCR、RMS、SC、SS、SF、LPCC、LSP、MFCC
SVM3	NFR、STE、SED、SF、LPCC、NRMSV、JRZM、4HME

4. Choice of effective samples

The training data points which are adjacent to the boundary between the two classes tend to support the decision boundary and can be chosen as effective samples for SVM training. The sample points which are far from the boundary are easy to be classified correctly and can be considered as approximate non-support vectors. In other words, by exploiting the spatial distribution of the samples, the whole training dataset can be portioned into disjoint clusters, each of which consists of either samples belonging to two classes (i.e. samples adjacent to the boundary) or samples belonging to only one class (i.e. approximate non-support vectors). Next, the new family of training data is constructed by choosing the clusters containing samples with different labels and replacing the clusters containing only non-support vectors by representative. The idea is essentially to eliminate data points that are not support vectors. A method using regression tree to accomplish this goal is proposed in this paper.

Regression trees are attractive due to their simplicity in model interpretation, and are particularly suited for effective data mining [10]. One of the important attributes of tree-based regression is its capability to generalize input-output mapping from the limited set of training samples. A regression tree is a binary tree constructed by repeatedly splitting (sub)sets of learning cases into two descendant subsets. Each node of a tree contains a subset of cases. A node that does not have descendant nodes is a terminal node. The root node comprises the entire samples. The left and right child nodes contain disjoint subsets of the parent content and are defined by splitting the parent node. The steps of choice procedure will be discussed in more detail in the following sections.

4.1. Regression tree building

In this paper, the regression tree is built by using LBG algorithm and K-means algorithm. The algorithm is formally implemented by the following recursive procedure:

1. Initiation: Design a 1-vector codebook as the root node; this is the centroid of the entire set of training vectors.
2. Splitting: Double the size of codebook by splitting each current codebook (parent node) y_n according to the rule (4), where n varies from 1 to the current size of codebook, and ε is a splitting parameter:

$$y_n^{+}=y_n(1+\varepsilon)\ ;\ y_n^{-}=y_n(1-\varepsilon) \qquad (4)$$

3. Clustering: Beginning with the new codebook, split each parent node into two child nodes using K-means clustering algorithm described in the following recursive process.
   a) Nearest-Neighbor Search: For each training vector, find the codeword in the current codebook that is closest in terms of similarity measurement, and assign that vector to the corresponding node.
   b) Mean update: Update the mean in each node using the centroid of the training vectors assigned to that node.
   c) Repeat steps a) and b) until the average distance falls below a present threshold.
4. Repeat step 2, 3 until the stopping criteria are met.

Two aspects should be pointed out for the algorithm. First, in step 3, we assign the vector to the subset in terms of Mahalanobis distance that takes into account not only the average value but also its variance and the covariance of the variables measured. Secondly, as mentioned in step 4, the tree building process goes on until some criteria are met. The process is stopped: (1) there is only one sample in each of the child nodes; (2) the farthest distance among all samples within each child node falls below a present threshold; or (3) the process has reached the limit on the number of levels in the maximal tree predefined according to the given task.

Now it is reasonable to assume that the regression tree corresponds to the acoustic space of the training data and descript the spatial distribution of the training data. Each node is assigned an acoustic class represented by the mean vector, covariance matrix and mixture weight, while every level can be viewed as a Gaussian Mixture Model (GMM). Next we will choose the effective samples within a proper tree level.

4.2. Effective samples choice

The deep level models contain too many clusters, which will remove too many predicted support vectors and can also result in decrease in prediction accuracy. The few clusters can not accurately model the distinguishing characteristics of the training set distribution. In this step, we will select the number of clusters contained within a certain level based on the Bayesian Information Criterion (BIC) according to the training data. The BIC of the GMM is formulated with the following function (5), where $\log P(X \mid \lambda)$ is a log likelihood of the training data X by the GMM when the number of nodes is M, d is the dimension of the acoustic feature, N is the number of frames of the training data.

$$BIC = \log P(X \mid \lambda) - 0.5M(2d+1)\log N \qquad (5)$$

For the selected level, all nodes can be divided into two types: the ones consisting of samples with different class labels and the ones consisting of samples with the same labels. After replacing the latter with those representative mean vectors, a new training set for a given SVM classifier can be obtained by collecting all node clusters within the selected level. On the other hand, we also can control the number of training sample points according to the distributions of different levels.

4.3. Handling unbalanced data condition

We discovered that the above steps tend to produce a final training set with very different numbers of samples for both classes. Specific step to alleviate this problem is required to balance the number of training points of binary classes. There are two ways to balance the data points for positive and negative classes: removing data points from the larger class and adding data points to the smaller class. We chose the latter in order to prevent information loss from the lager class, as well as to add information to the smaller class. We added data points from the clusters which had already been replaced by the centroid vectors in the smaller class to achieve balance.

5. Experimental results

5.1. Dataset and experimental condition

The data used in our experiments are collected from real TV programs, which are about 343 minutes in total. 94 minutes of data are used for training, and 249 minutes of data are used for testing. The training set consists of 25 minutes of pure speech, 25 minutes of non-pure speech, 25 minutes of music and 19 minutes of environment noise. The test set includes 109 minutes of pure speech, 103 minutes of music, 25 minutes of non-pure speech and 12 minutes of environment noise. Pure speech and non-pure speech can be combined into speech class, while music and environment noise can be combined into non-speech class.

The training and testing data are all converted into the uniform format of 8-KHz, 16-bit, mono-channel. In our experiments, we set 1s as a test unit. If there are two audio types in a 1s audio clip, we will classify it as the time-dominant audio type. For LIBSVM, we perform a grid-search on C and γ using cross-validation. We try exponentially growing sequences for pairs of (C, γ) ($C = 2^{-1}, 2^{0}, \cdots, 2^{13}$; $\gamma = 2^{-4}, 2^{-3}, \cdots, 2^{3}$) and the one with the best cross-validation accuracy is picked as the optimal parameters.

5.2. Result and analysis of effective samples choice

The columns in Table 2 contain the audio types, the size of the original training set used for the experiments (nSample), the number of removal redundant data based on the proposed method (Removal). For SVM1 and SVM3, we could obtain a relatively small and balanced number of training data points according to our choice process. Furthermore, only about 10 percent of support vectors trained with the whole original training set are removed by our proposed samples choice method. For SVM2, since the ratio between the number of support vectors and the total number of training data points is high, we can predict the data is relatively highly unseparable. This means that the data set does not contain many data points that have similar information, so we can not achieve effective data reduction using the proposed method.

Table 2: *Results of effective samples choice*

	Audio types	nSample	Removal
SVM1	Speech	2918	1019
	Non-speech	2628	787
SVM2	Pure speech	1490	117
	Non-pure speech	1428	95
SVM3	Music	1547	891
	Noise	1081	396

5.3. Performance evaluation

Table 3: *Training result based on the original training set*

	C	γ	nSV	Accuracy	Time(h)
SVM1	194	0.71	1536	98.29%	10.33
SVM2	14263	0.37	1000	97.81%	1.37
SVM3	64	0.19	371	98.21%	0.70

Table 4: *Training result based on the new training set*

	C	γ	nSV	Accuracy	Time(h)
SVM1	446	0.35	809	98.36%	3.20
SVM2	4.50	0.38	989	97.78%	1.10
SVM3	337	0.31	261	96.72%	0.22

Table 5: *Comparison of testing classification performance*

	Non-pure speech	Pure speech	Noise	Music
Original data	91.10%	98.78%	96.56%	95.73%
Effective samples	91.51%	99.04%	96.70%	95.50%

The training and testing results based on the original and the new training set are shown in Table 3, 4 and 5, respectively. The columns in Table 3 and 4 contain the classifier name, the optimal value of the parameters (C,γ), the total number of SVs (nSV), the training cross-validation accuracy and training time (hours). Our experiments were run on a PC which utilizes a Pentium4 2.4GHz processor and a maximum of 512MB of memory. From the experiments, it can be seen that the running time of the proposed method is greatly shorter than that of the original method, and the training accuracy and the testing accuracy of the presented method are almost the same as those of the original method. On the other hand, we should notice that our algorithm uses fewer support vectors and keeps good generalization performance.

Generally speaking, while the training time dramatically decreased and the number of support vectors decreased a small amount, we can say that our algorithm has chosen very compact data points that maintain the original classification performance.

6. Discussion and Conclusions

In this paper, a choice algorithm of effective samples for fast SVM training has been presented in audio classification system. We can observe the fact that the optimal solution still holds if any non-support vector is removed. Moreover, support vectors actually constitute only a small fraction of the training samples. If most non-support vectors can be removed quickly at the first step, SVM training can be accelerated dramatically. Under the above consideration, we implement the process of effective samples choice based on regression tree by exploiting the distributional properties of the training data, that is, the natural clustering of the training data and the overall layout of these clusters relative to the decision boundary of support vector machines. Experimental results show that our proposed method dramatically improves the speed of SVM training without reducing the generalization performance of SVM.

7. Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (NSFC) under the grant No. 60475014.

8. References

[1] Omar, A. H., “Audio Segmentation and Classification”, Master's thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, 2005.

[2] Lu, L., Zhang, H. J. and Li, S., “Content-based audio classification and segmentation by using support vector machines”, ACM Multimedia Systems Journal, 8 (6), pp. 482-492, March, 2003.

[3] Platt, J., “Sequential minimal optimization: A fast algorithm for training support vector machines”, in advances in Kernel Methods - Support Vector Learning, 1998, pp. 185--208.

[4] Chang, C. C. and Lin, C.J., “LIBSVM: a library for support vector machines”, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[5] Lebrun, G., Charrier, C. and Cardot, H., “SVM Training Time Reduction using Vector Quantization”, ICPR (1) 2004: 160-163.

[6] Bakir, G.H., Bottou, L. and Weston, J., “Breaking SVM Complexity with Cross-Training”, NIPS2004.

[7] Yang, X. W., Lin, D. Y., Hao, Z. F., Liang, Y. C., Liu, G. R. and Han, X., “A fast SVM training algorithm based on the set segmentation and *k*-means clustering”, Progress in Natural Science, Vol. 13, no. 10, pp. 750-755, 2003.

[8] Boley D. and Cao D. W., “Training support vector machine using adaptive clustering”, Proc. of Fourth SIAM International Conference on Data Mining, Lake Buena Vista, FL, United States, 2004. 126 - 137.

[9] Panagiotakis, C. and Tziritas, G., “A speech/music discriminator based on RMS and zero-crossings”, IEEE Transactions on Multimedia, Vol. 7, No. 1, Feb. 2005.

[10] Breiman, L., Friedman, J. et al., “Classification and Regression trees”, Wadsworth, Belmont, CA, 1984.