MIREX 2012 SUBMISSION
AUDIO CLASSIFICATION USING SPARSE FEATURE LEARNING

Juhan Nam
CCRMA
Stanford University
juhan@ccrma.stanford.edu

Jorge Herrera
CCRMA
Stanford University
jorgeh@ccrma.stanford.edu

ABSTRACT

We present a training/test framework for automatic audio annotation and ranking using learned feature representations. Commonly used audio features in audio classification, such as MFCC and chroma, have been developed based on acoustic knowledge. As an alternative, there is increasing interest in learning features from data using unsupervised learning algorithms. In this work, we apply sparse Restricted Boltzmann Machine to audio data, particularly focusing on learning high-dimensional sparse feature representation. Our evaluation results on two music genre datasets show that the learned feature representations achieve high accuracy.

1. INTRODUCTION

Content-based Music Information Retrieval (MIR) tasks take the audio content in the form of features. Most popularly used audio features, such as MFCC, chroma and spectral summaries (centroid, flux, roll-off, etc.) are extracted in a succinct form, which has been engineered based on acoustic knowledge. While these engineered features have been successfully applied to many of MIR tasks, there is increasing interest in learning the features adaptively to data. The learning algorithms capture underlying structures of data in an unsupervised manner and represent the feature given the learned structure.

In our previous work, we presented a data processing pipeline to learn features from audio signals [3]. We showed the suggested method outperforms prior arts in music annotation and retrieval. In this work, we apply the same feature representation method to a single-label audio classification, particularly focusing sparse Restricted Boltzmann Machine (RBM) [2]. The sparse RBM was proven to be superior to other compared algorithms in our previous work. Our evaluation results on the Cal500 [4] and Magnatagatune [1] datasets show that the proposed method achieves high accuracy using a simple classifier.

2. PROPOSED METHOD

The proposed data processing pipeline to train data is composed of several stages. Here we summarize each stage briefly. They are detailed in [3].

2.1 Preprocessing

We chose mel-frequency spectrogram as input data for RBM. In order to reduce amplitude variations in the spectral domain, we first applied time-frequency automatic gain control to raw audio. After computing mel-frequency spectrogram, we compressed the amplitude using an approximated log scale.

2.2 Feature Representation By Learning

Feature learning algorithms need to take a specific size of input units where data dependency is captured. We took multiple consecutive frames of mel-frequency spectrogram as an input unit so that the algorithm can learn not only timbral but also short-term temporal dependency. Before feeding the data into the RBM, we applied PCA whitening as another preprocessing step. This enables fast and effective feature learning by reducing data dimensionality and removing the pair-wise correlation in the input data. We set the RBM to produce high-dimensional sparse feature vectors. After training the RBM, we extracted the local feature vectors in a convolutional manner and then, by aggregating them, formed a song-level feature vector where a label is given. Before the aggregation, we performed max-pooling over one second or so (assuming that the length of song files is around 30 second).

2.3 Classification

The song-level feature vectors and corresponding tags are used for supervised training. We train multiple linear SVMs (as many as tags are present in the dataset). For the classification task (binary output), we use the SVM boundary to discretize the estimates. For the ranking (affinity) task, the SVM outputs are used directly.

3. EXPERIMENTS AND RESULTS

3.1 Datasets

We evaluated the proposed method on two datasets. The first one is“sampled” version of the Cal500 dataset. Multi-

	Task (Metric)	
Dataset	Ranking (AROC)	Annotation (F-score)
Cal500	0.8535	0.5107
Magnatagatune	0.7244	0.1123

Table 1: Best classification AROC (ranking) and F-score (annotation) on the test subsets for the "sampled" Cal500 and the Magnatagatune datasets.

ple ten second snippets—60 seconds apart—were extracted from each song in the original dataset. The song-level tags were assumed to apply to each snippet in the song. This "sampled" dataset was split into training (80%) and testing (20%) subsets. The second dataset used was the Magnatagatune dataset. Similar to what we did in the Cal500 case, we used only 10 seconds of each song, although in this case a single 10-second snippet was used, due to the large amount of songs in the dataset. Also similarly to what we did in the Cal500, we used separate training and testing subsets.

The models were trained using the training subset, running internal 3-fold cross validation for optimal parameter estimation. Different models were trained for each task (annotation and ranking) and were then evaluated on the corresponding test subset.

3.2 Parameters

We first resampled the waveform data to 22.05kHz and applied the time-frequency AGC using 10 sub-bands and attack/delay smoothing the envelope on each band. We computed an FFT with a 46ms Hann window and 50% overlap, which produces a 513 dimensional vector (up to half the sampling rate) for each frame. We then converted it to a mel-frequency spectrogram with 128 bins. For PCA whitening and feature learning steps, we sampled 100,000 data examples, approximately 200 examples at random positions within each song. Each example is selected as a 128×4 patch from the mel-frequency spectrogram. In PCA whitening, we retained 90% of the variance and added 0.01 to the variance for regularization.

For sparse RBM, we used dictionary size (or hidden layer size) and sparsity as the primary feature-learning parameters. The dictionary size was fixed to 1024 and different sparsity parameter values were tested—0.01, 0.02, 0.03 and 0.05—picking the one that produced the best accuracy. Max-pooling was performed over segments of length 0.5, 1, 2 and 4 seconds, again selecting the one producing the best accuracy.

3.3 Results and Discussion

The results are summarized in Table 1. They are comparable to those reported by state-of-the-art music tagging algorithms. Note that we used a simple linear SVM classifier. This indicates that different tags are linearly separable in the learned feature space.

4. REFERENCES

[1] Edith Law and Luis von Ahn. Input-agreement: A new mechanism for data collection using human computation games. In *CHI 2009*, 2009.

[2] H. Lee, C. Ekanadham, and A. Ng. Sparse deep belief net model for visual area v2. *Advances in Neural Information Processing Systems*, 2007.

[3] J. Nam, J. Herrera, M. Slaney, and J. Smith. Learning sparse feature representations for music annotation and retrieval. In *ISMIR*, 2012.

[4] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet. Towards musical query-by-semantic description using the CAL500 data set. In *ACM Special Interest Group on Information Retrieval Conference*, 2007.