MULTI-TIMESCALE PMSCS FOR MUSIC AUDIO CLASSIFICATION

Philippe Hamel
hamelphi@iro.umontreal.ca

1. INTRODUCTION

This extended abstract describes a system that uses multi timescale Principal Mel-Spectrum Components and a combination of temporal pooling functions to solve the task of music audio classification and tagging. These concepts were introduced in [3] and [2]. We present here technical details and a summarized description of the model. For a deeper explanation, see the original papers.

1.1 Feature Extraction

Principal mel-spectrum components (PMSCs) [3] are computed several timescales in parallel [2]. For each timescale, the feature extraction involves four steps: discrete Fourier transform (DFT), mel-scaling, principal component analysis whitening (PCA) and temporal pooling.

Firstly, for each timescale, we compute discrete Fourier transforms over a given time length. To compute PMSCs at different timescales, we simply use different time length for the DFT. In this system, we use a combination of 5 timescales: 46ms, 93ms, 186ms, 372ms and 743ms. We use the same frame step of 23ms for all timescales, meaning that there is more overlap for longer timescales.

Secondly, we run the spectral amplitudes of the DFTs through a set of 200 mel-scaled triangular filters to obtain a set of spectral energy bands. We take the logarithm of the amplitude of those bands.

Then, we compute the principal components of a random sub-sample of the dataset (roughly 60,000 frames). In order to obtain features with unitary variance, we multiply each component by the inverse square of its eigenvalue, a transformation known as PCA whitening. Here, the goal of the PCA is to diagonalize the covariance matrix, not to reduce dimensionality. Thus, we keep all the principal components (yielding 200 dimensions per timescale). The PCA whitened mel-scaled energy bands are referred to as PMSCs.

In the next step, we apply temporal pooling, i.e. we compute statistics over the PMSCs over a given window length. We chose a time window of roughly 1.5 seconds. Within this time window, we compute the mean, the variance, the minimum and the maximum of each feature. Thus, for each timescale and for each 1.5 seconds window, we obtain $4 * 200 = 800$ features.

We then concatenate all the timescales in a single vector corresponding to a window of 1.5s, yielding $800*5 = 4000$ features per window.

2. MODEL

Each pooling window is considered as a training example for the classifier, and we average the predictions of the classifier over all the windows of a given clip to obtain the final classification. The classifier is a single hidden layer neural network, also known as multi-layer perceptron (MLP). We used a hidden layer of 2000 units, sigmoid activation, L2 weight decay and cross-entropy cost. The MLP was coded in python using the Theano library [1].

The neural network is trained by gradient descent. The full training set is divided in two subsets, a training set and validation set. The network is trained by using mini-batches of 10 examples (10 full excerpts) at a time.

We measure the AUC as our validation measure. Using this validation measure we use early-stopping to know when to stop training the network.

3. CLASSIFICATION

The MLP outputs an affinity prediction for each class. These predictions are done over excerpts of roughly 1.5 seconds. The predictions are averaged over all the windows from one song to obtain a prediction for the whole song. For the audio classification task, we simply choose the class with the highest activation at the output of the MLP as the predicted class.

4. TAGGING

There are two required outputs for the audio tag classification : affinity and binary classification.

4.1 Affinity

The MLP already outputs an affinity prediction between 0 and 1 for each possible tag. The affinity scores for a song is thus directly the averaged output of the MLP over all windows.

4.2 Binary Classification

For the binary classification, we need to define a threshold at which to discriminate between an positive and negative tags. We choose the threshold that optimizes the F1-score on the validation set. Thus, all the tags with an affinity higher than the chosen threshold will be considered as positive tags.

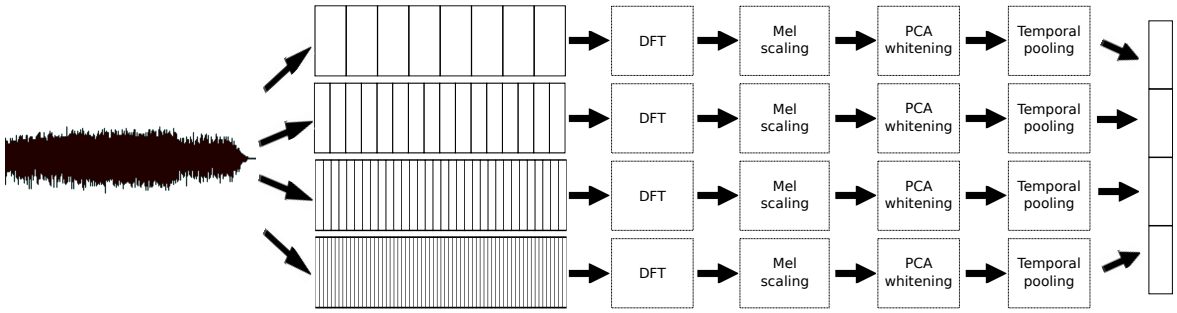

Figure 1. PMSCs are computed in parallel at different timescales.

5. REFERENCES

[1] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010.

[2] P. Hamel, Y. Bengio, and D. Eck. Building musically-relevant audio features through multiple timescale representations. In *Proceedings of the 13th International Conference on Music Information Retrieval (ISMIR'12)*, Porto, Portugal, 2012.

[3] P. Hamel, S. Lemieux, Y. Bengio, and D. Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR'11)*, 2011.

[4] E. Law and L. von Ahn. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the International Conference on Human factors in computing systems*, CHI. ACM, 2009.