POOLED FEATURES CLASSIFICATION MIREX 2011 SUBMISSION

Philippe Hamel
hamelphi@iro.umontreal.ca

ABSTRACT

This extended abstract describes a system that uses Principal Mel-Spectrum Components and a combination of temporal pooling functions to solve the task of music audio classification and tagging.

1. INTRODUCTION

This extended abstract presents a model that was introduced in [2] as the Pooled Features Classifier (PFC). We present here a summarized description of the model. For a deeper explanation, see the original paper.

1.1 Audio Preprocessing

Our audio preprocessing involves three steps: discrete Fourier transform (DFT), mel-compression and principal component analysis whitening (PCA).

Firstly, to transform the audio in the spectral domain, we compute DFTs over windows of 1024 samples on audio at 22.05 KHz (i.e. roughly 46ms) with a frame step of 512 samples. Then, we run the spectral amplitudes through a set of 256 mel-scaled triangular filters to obtain a set of spectral energy bands. We compute the principal components of a random sub-sample of the training set. 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. We will refer to the preprocessed audio features as Principal Mel-Spectrum Components (PMSC).

2. MODEL

The model applies a given set of pooling functions to the PMSC features, and sends the pooled features to a classifier. 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. We will refer to this model as the Pooled Features Classifier (PFC) model.

3. CLASSIFICATION

The MLP outputs an affinity prediction for each class. 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 output of the MLP.

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.

5. ACKNOWLEDGMENTS

The author would like to thank Simon Lemieux, Yoshua Bengio and Douglas Eck. The author is also thankful towards the Theano [1] development team. This research was financed in part by FQRNT.

6. 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. Oral.

[2] Yoshua Bengio Philippe Hamel, Simon Lemieux and Douglas Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *ISMIR*, 2011.