CNN-BASED AUDIO ONSET DETECTION MIREX SUBMISSION

Jan Schlüter
Austrian Research Institute for
Artificial Intelligence, Vienna, Austria
jan.schlueter@ofai.at

Sebastian Böck
Department of Computational Perception,
Johannes Kepler University, Linz, Austria
sebastian.boeck@jku.at

ABSTRACT

This submission to the MIREX 2013 Audio Onset Detection task uses a trained Convolutional Neural Network (CNN) to detect likely positions of onsets in an audio signal. It is based on our work described in [2].

1. INTRODUCTION

Convolutional Neural Networks (CNNs) process input data by convolving it with filters that are optimized to produce given target outputs. In [2], we trained a CNN on annotated spectrograms to predict likely musical onsets, and obtain performance comparable to the state of the art in onset detection. Our MIREX submission builds on this work, using the exact same network architecture, but a better training procedure.

2. METHOD

In the following, we will describe our method in detail.

2.1 Feature Extraction

From the 44.1 kHz mono input signal, we compute three Hann-windowed STFTs with a frame size of 1024, 2048 and 4096 samples, respectively, and a hop size of 441 samples (i.e., a frame rate of 100 Hz). We take the absolute values of the spectra and apply a filterbank with 80 triangular normalized filters spaced linearly on a mel-scale, with the lowest filter starting at 27.5 Hz and the highest one ending at 16 kHz. We take the element-wise natural logarithm.

2.2 Network Architecture

Our CNN is fed with input blocks of 15 frames by 80 bands by 3 channels (the 3 different STFTs). The first layer is convolutional, with 10 filters of 7 frames by 3 bands by 3 channels, resulting in a feature map of 9 frames by 78 bands by 10 channels. The second layer performs max-pooling over 3 bands without overlap, reducing the feature map to 9 frames by 26 bands by 10 channels. We add a scalar bias term per channel and apply the tanh nonlinearity. The third layer is convolutional again, with 20 filters of 3 frames by 3 bands by 10 channels, followed by another max-pooling over 3 bands, followed by bias terms and tanh nonlinearity. The fifth layer is a fully-connected layer of 256 sigmoid units, and the final layer is fully connected with a single sigmoid output unit to predict onsets. The architecture is depicted in Figure 1.

2.3 Network Training

The network is trained on a dataset of 401 annotated audio files totalling in about 110 minutes and 27,000 onsets (the same dataset as used for the submissions of Sebastian Böck in previous years). Training is performed in mini-batches of 256 samples, for 300 epochs, using gradient descent with momentum. We use an initial learning rate of 1.0, multiplied by 0.995 in each epoch, and an initial momentum of 0.45, linearly increased to 0.9 between epochs 10 and 20. As opposed to our previous work, for each training case we randomly drop 50% of the inputs of the two fully-connected layers and double the remaining connection weights, to improve generalization and avoid the need for early stopping (see [1]). As another novelty, any frame directly preceding or directly following an annotated onset frame is treated as a positive example rather than a negative example, but weighted with only 25% in training.

2.4 Network Predictions

To detect onsets on new data, features could be fed to the trained network in blocks of 15 frames, recording the network output for each instance to obtain an onset activation function over time. However, we can use the fact that shifting the input by one frame shifts all feature maps by one frame (due to the definition of the convolution operation and because we never pool over time): Computing the onset activation function from the multi-channel spectrogram of a file can efficiently be implemented with three convolutions (for the first, third and fifth layer), two max-pooling operations over frequency bands (for the second and forth layer), one dot product (for the final layer), and four element-wise nonlinearities. As opposed to the training phase, no inputs are dropped in the fully-connected layers when computing predictions.

The obtained onset activation function is smoothed by convolution with a Hamming window of 5 frames. Frames with a higher activation than both their predecessor and successor are onset candidates. Candidates with a higher activation than a given threshold are reported as onsets.

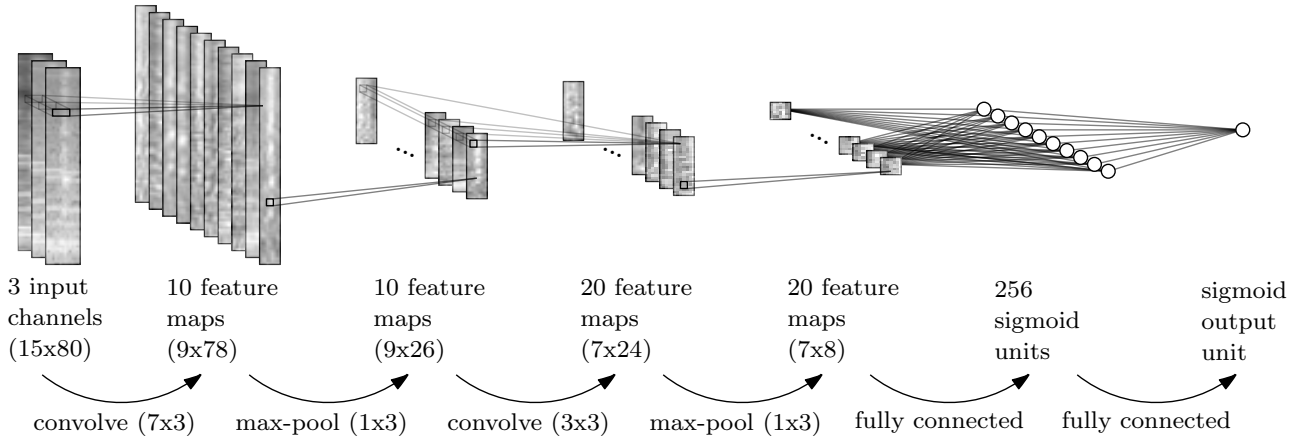

Figure 1. The Convolutional Neural Network architecture used in this work. Starting from a stack of three spectrogram excerpts, convolution and max-pooling in turns compute a set of 20 feature maps classified with a fully-connected network.

3. RESULTS

We take a brief look at the efficiency and the accuracy of our algorithm.

3.1 Computational Efficiency

On a single core of a 3.4 GHz i7-2660 CPU, feature extraction in numpy/scipy takes about 2 s for 60 s of input data, and applying the CNN takes 5.5 s in numpy/scipy, 1.6 s in theano using the CPU and 0.12 s in theano using a GTX 480 GPU. We estimate a production-grade implementation to perform at 15x realtime on CPU and at least 50x realtime on GPU, depending on the gain of moving the feature extraction to GPU.

3.2 Detection Accuracy

(Filled in when results become available.)

4. ACKNOWLEDGEMENTS

This research is supported by the Austrian Science Fund (FWF): TRP 307-N23, and by the European Union Seventh Framework Programme FP7 / 2007-2013 through the PHENICX project (grant agreement no. 601166). The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Transport, Innovation, and Technology.

5. REFERENCES

[1] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R.R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. arXiv, 2012.

[2] Jan Schlüter and Sebastian Böck. Musical Onset Detection with Convolutional Neural Networks. In *6th International Workshop on Machine Learning and Music (MML), In conjunction with the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD)*, Prague, Czech Republic, 2013.