MUSIC TRANSCRIPTION WITH A CONVOLUTIONAL NEURAL NETWORK 2016

Daylin Troxel
Lunaverus
daylin.troxel@lunaverus.com

ABSTRACT

In this submission for MIREX 2016 I give an overview of a method for using a convolutional neural network to identify notes in music. Spectrograms were first generated from audio using a constant Q transform and possible note onset times were detected. Regions of the spectrograms centered at these onset times were fed as input into a convolutional neural network, which produced a vector of probabilities for 88 notes. Filtering was performed on these probabilities to obtain the final note predictions.

1. INTRODUCTION

Convolutional neural networks (CNNs) have been used for many years to achieve state-of-the-art results in computer vision. [1] Finding notes in a region of a spectrogram can be viewed as an image detection problem with a few differences. CNNs often to use a final softmax to pick the most likely class (car, dog, etc.) for an input image among a fixed number of competing classes. For polyphonic note detection, it's desirable to remove this layer to allow multiple outputs to be active at once. Spectrograms are simpler than most photographic images in the sense that spectrograms can be approximately composed of only a couple basic structures: harmonics, which are narrow in frequency and wide in time, and drums or other wideband features that are narrow in time and wide in frequency. Unlike objects in photographs, notes in a spectrogram are never subject to rotation or distance-based scaling. Ignoring drums, notes form the only object class a neural network needs to learn to recognize to perform useful transcription. If the spectrogram uses a logarithmic scale on the frequency axis, all notes will have the same spacing between the 1st and 2nd harmonics, the 2nd and 3rd, and so on. Notes with different pitches become translated versions of the same image with some variation in harmonic amplitudes. Convolutions are ideal to handle this type of translation. Weight sharing insures that any improvement in the ability of the network to recognize one note improves its ability to recognize notes at every other frequency.

One of the ways in which note detection is uniquely challenging is the way nearby harmonics interfere with each other. In photographs, one object may be partially hidden by another, but placing object 1 behind object 2 doesn't cause object 2 to become distorted where they overlap. One way to reduce this distortion is to dynamically detect collisions during the spectrogram generation and increase the Q-factor where they occur. A larger Q factor comes at the cost of reduced amplitude information, so a low Q is preferred by default.

Neural networks can be trained by providing examples of inputs with the desired outputs and using backpropagation to iteratively update the weights of the network. The weights are moved down the gradient of the error such that the desired outputs become more likely. The more training data that is available, the larger and more accurate the neural network can be. Fortunately, it's easier to generate large quantities of labeled note data than it is to label images. MIDI files are effectively pre-labeled data when coupled with a good MIDI-to-WAV utility and large collections of MIDI files are freely available. The CNN in this submission used around 2.5 million labeled samples auto-generated from a set of 3,000 MIDI files.

2. ALGORITHMS

Stereo spectrograms were generated for both channels of audio using an approximately-constant Q factor that increased as needed to reduce harmonic collisions. Additional non-linear scaling was applied to make the amplitudes more closely match the log of raw audio amplitudes and to add some normalization. The number of frequency bins per note was chosen to be an integer so that after a number of max-pooling and size reduction layers there would be exactly one output neuron per note. The stereo spectrograms were stacked in a 3-dimensional array, analagous to color channels in an image.

Candidate note start times were detected by looking for regions in the spectrogram where the amplitude of many frequency bins increased in a short period of time. Rectangular slices of the spectrogram centered on these time values were used as inputs to a CNN. The areas of the spectrogram before and after the detected onset time provide additional context helpful in detecting notes.

The CNN's architecture was based on Microsoft's ResNet, which won the ILSVRC challenge in 2015 [2]. The forward skip connections speed up the training process and allow the output to be composed of a series of additions (residuals) on the input. This is an attractive choice for note detection, because the output of the CNN generally looks like the input after subtracting off the 2nd and above harmonics and possibly increasing the amplitude of the fundamental frequencies. In contrast to ResNet, this CNN made use of several pairs of long, thin convolutions with alternating vertical and horizontal orientations. These long, thin convolution pairs, an Mx1 followed by a 1xN, helped efficiently integrate information from distant parts of the spectrogram and may have helped resolve ambiguities in fundamental frequencies by connecting many neurons along the frequency axis. The final output layer was a 1x1 convolution with a single filter followed by a sigmoid that resulted in an 88x1x1 array of note probabilities.

Additional filtering was applied on the output of the CNN, which resulted in the removal of some of the lower confidence notes (notes with greater than 0.5 probability but less than some threshold). This filtering consisted of a separate note detection algorithm, which used a more traditional approach of searching for peaks in the spectrogram, forming tracks of sequential peaks, and ordering candidate notes from most to least likely. The main idea behind this secondary algorithm is that the strongest track at any instant is very likely to be a low harmonic: a 1st, 2nd, or 3rd. The likelihood of each of these possibilities can be ranked using the scores generated by the CNN. The most likely fundamental frequency is selected and all tracks at multiples of this frequency are removed from consideration. The process is then repeated until there are no more tracks with strong amplitudes remaining. This algorithm produces a second semi-independent set of notes based on strong amplitude tracks in the spectrogram and scores from the CNN. To obtain the final set of notes at a given moment in time, all of the high-confidence notes from the CNN were accepted, but all of the notes with lower confidences were filtered out unless they also appeared in the second list of track-based notes.

For simplicity, no attempt was made to accurately determine note offsets. A note was assumed to end as soon as the next note began.

3. EVALUATING PERFORMANCE

The CNN had an accuracy of 99.088% on the evaluation split (data not used for training) at the note level after one epoch of training. This was calculated by rounding each output to 0 or 1 and measuring the fraction of all outputs that matched the truth values. However, since the dataset had an average of 3 notes and 85 non-notes per training example, an unintelligent algorithm achieves 96.6% note level accuracy by never detecting any notes. At the frame level, classifying an entire frame as accurate if all 88 outputs match their truth values, the accuracy was just over 50%. The accuracy of the entire algorithm, which depends on the onset time detection accuracy and the filtering of the CNN's output has not yet been determined, but will be published in the MIREX 2016 results. The performance on non-MIDI-based music has not been objectively measured, but in informal observations the accuracy seems to be roughly comparable.

One potential area for improving this algorithm is to use a recurrent neural network (RNN) to learn common sequences and combinations of notes and adjust note scores based on these expectations. Some sequences and combinations of notes are statistically more likely than others and this information could be used to resolve ambiguities. An RNN may also be well-suited for formatting notes into measures and generating sheet music that is easier for humans to read. Collections of sheet music in a standard format like MusicXML could be used to generate a training data set for this type of RNN.

4. REFERENCES

[1] J. Schmidhuber. Deep learning in neural networks: An overview. Neural Networks, 2014.

[2] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)