Sound Classification with Artificial Neural Networks

Noah Black

Goal

- First, I want to train a neural network to classify a drum sound as either a 'hi-hat', a 'snare', or a 'kick drum'.
- Then, I want to write a script that uses the network to identify and export individual drum sounds from a drum recording.

Deciding how to prepare digital audio as input for an ANN

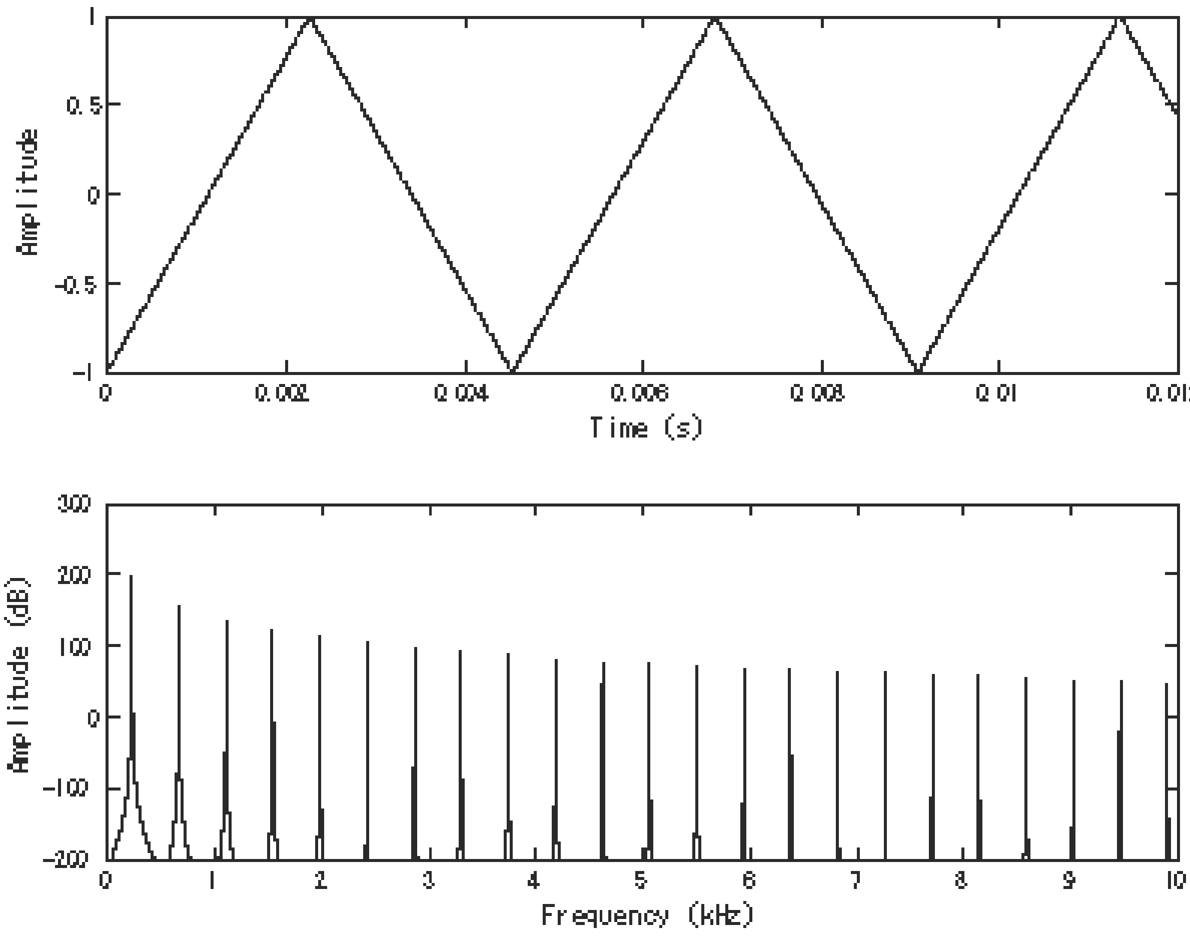
- Which properties of a drum sound can be used to distinguish it from other drum sounds?
  - Frequency content (lows, highs, mids)?
    - Sounds important
  - Attack and decay of the sound?
    - Sounds helpful, but how to capture this in a few scalars?
  - Pitch (can be determined from frequency content)
    - Probably not necessary. A particular kind of drum isn't likely to be characterized by a certain pitch
  - Loudness
    - Sounds helpful

Preparing a sound file for the network

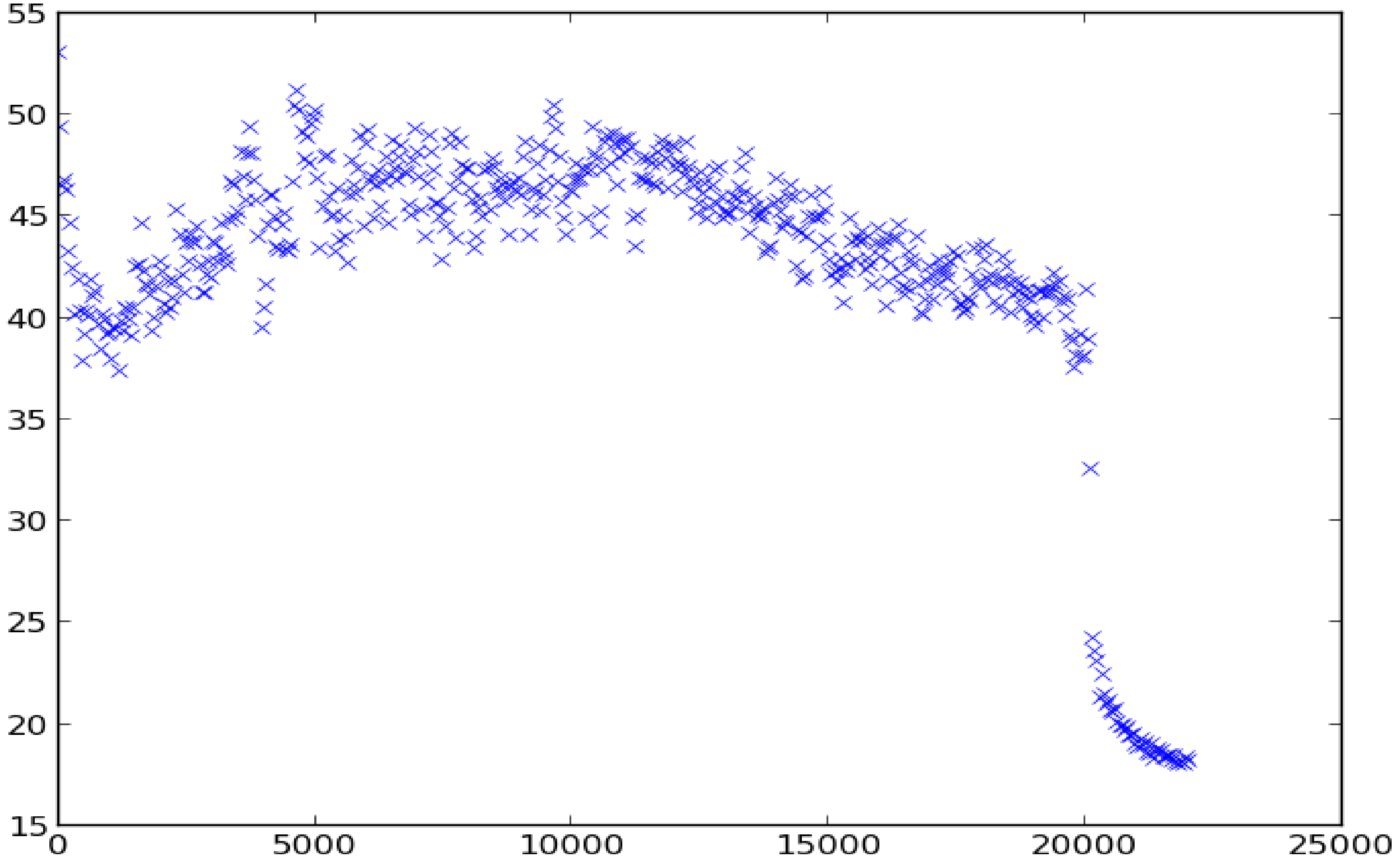
STEP 1: GENERATE FREQUENCY SPECTRUM

- Divide sequence of audio samples into equally-sized frames of samples, z-transform the frames, average the transforms
- This gives you an array of average magnitudes of frequency bands.
- Freq spectrum sort of like Winamp EQ
- Frequency bands are network inputs
- I experimented with several spectral resolutions
  - 256 was most successful
- Each band is normalized according to the range of that band across the training dataset

Frequency spectrum of triangle wave oscillation



Frequency Spectrum of Hi-hat Recording in MATLAB



Preparing a sound file for the network

STEP 2: CALCULATE THE AVERAGE OF THE DERIVATIVE OF THE SOUND'S SAMPLES SQUARED

- This produces a scalar that should (?) be negative when a signal is decaying.
- This was improvised (I'm not really a math guy) and I suspect it's not the best way to measure the way a recording attacks/decays, but it seemed to improve the results.
  - Drum sounds that did not begin with a sharp attack (as drums usually do) were not given a high score in any category (helpful for part 2)

Preparing a sound file for the network

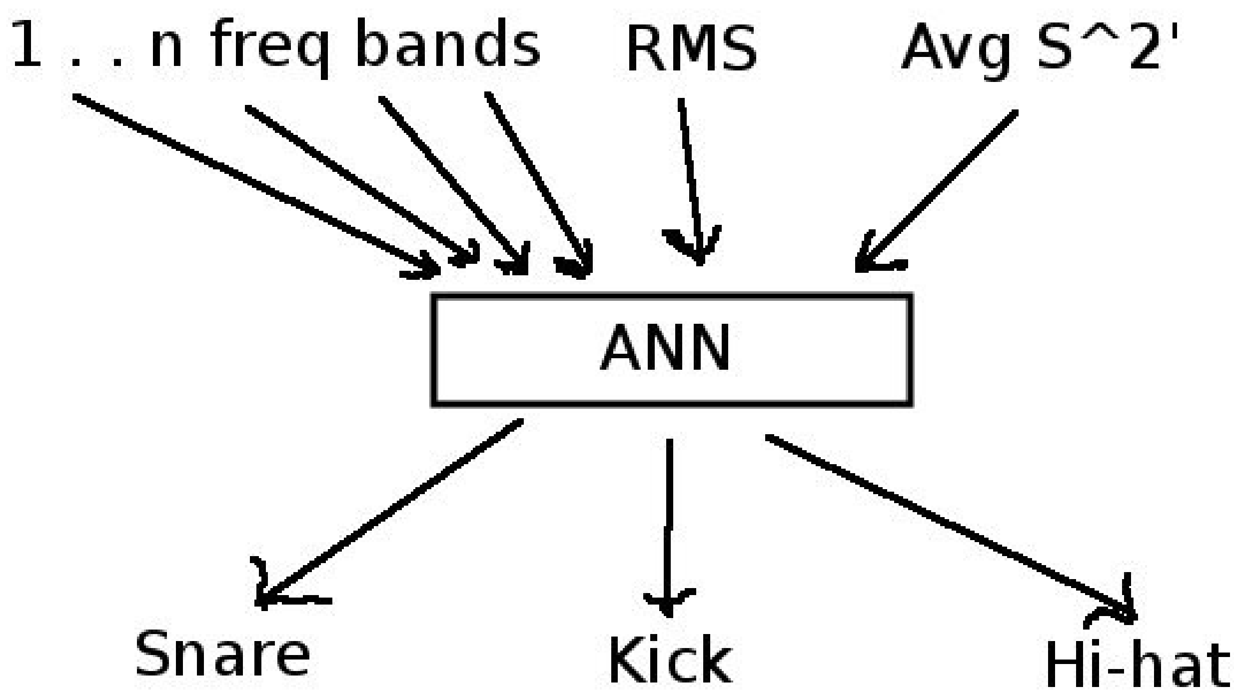
STEP 3: CALCULATE THE RMS OF THE SOUND

- Root mean square of samples
- This is of measure of loudness
- $$RMS = \sqrt{\frac{1}{n}\sum_n x^2(t)} = \sqrt{\frac{1}{n^2}\sum_n |X(f)|^2} = \sqrt{\sum_n \left|\frac{X(f)}{n}\right|^2}.$$

Preparing a dataset for training

- I used the PyBrain artificial neural network library to create and train a back-propagating neural network
  - Momentum: 0.2
  - Weight decay: 0.0001
  - N inputs, N/2 hidden neurons, and an output neuron for each class
- I downloaded a small library of drum sounds, which I organized into directories for each class (snare, hi-hat, kick)
- Then, each of these sounds was prepared for input as previously explained
- PyBrain provides methods for training a network with a user-provided data set of inputs and their corresponding expected outputs

Ugly diagram



Training the dataset

- Once a dataset is prepared, the database is trained for some number of epochs. I found that about 500 epochs was sufficient. This took about 5-20 minutes, depending on the spectral resolution

Using the Network

- Once the network is trained, it is serialized to a file
- The network can be loaded and used later
- The network can be used to classify drum sounds not present in the training dataset
- The network can be used to find drum sounds in a larger audio file

Identifying drum sounds in an audio file

- I wrote a script that scans through an audio file, and divides it into many overlapping frames
- Attempts to classify each of those frames
- Maintains the “best” snare, hi-hat, and kick found
- Writes the best drum sounds to a file

Other Possible Applications

- Pitch detection?
  - I tried this – didn't work at all. The dataset could not even be successfully trained.
  - This might work with a very high spectral resolution, or a smarter developer
- Auto-organizing a library of drum sounds
- Assembling a kit of drum sounds sampled from a piece of music
  - Hip hop producers do this manually
- Detecting noises in machinery that might indicate serious problems

Conclusion

- Neural networks can be quickly trained to perform basic classification of well-prepared audio data
  - Pitch detection is another story
- How to represent the data is the most important decision
- A larger, more diverse training dataset is likely to improve results