Music Transcription with Neural Networks

Zach Hinkle

April 23, 2018

Why Neural Nets?

- Each music note correspond to a unique fundamental frequency
- However, mixtures of overtones cause the problem to be too diffuse to be solved with traditional programming
- Good results have been produced transcribing music using convolutional neural networks with spectrograms as input
  - https://www.lunaverus.com/cnn
- Goal is to see if similar or better results can be produced using fft of audio files as input
  - A similar problem was performed as a master thesis about a year ago
  - used midi files; however, midis are easily converted to scores by music notation software like Finale or Sibelius

Data

- The data used in the master thesis is available on github:
  - Includes a large corpus of midi files with corresponding scores
  - Restricted to polyphonic piano music
- MAPS is a program dedicated to research in music transcription
  - http://www.tsi.telecom-paristech.fr/aao/en/2010/07/08/maps-database-a-piano-database-for-multipitch-estimation-and-automatic-transcription-of-music/
  - Has a large body of data (31 GB) with singles notes, standard chords, random chords (sometimes called "jazz"), and short clips of music in .wav format
  - all are tagged with the corresponding notes being played in text and midi format

Problem Formulation

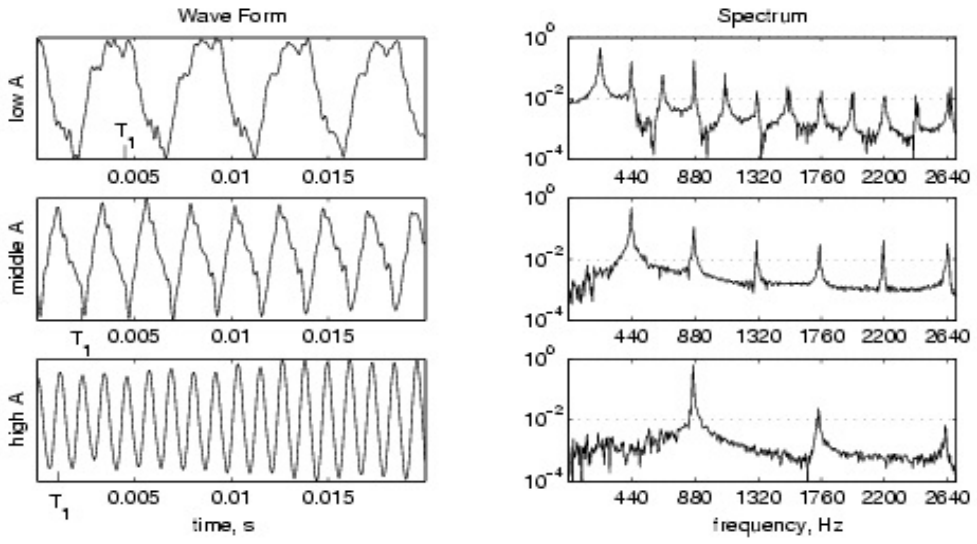

Figure: Typical musical waveforms in time and frequency domain

- Use fft of wav files of piano music as input
- Can be handled easily in python using the scipy.fftpack and scipy.io libraries
- Either text or midi file as output

Problem Formulation

- can use the python extension abjad to convert text into LaTeX markup, then use LaTeX extension lilypad to produce scores
  - Abjad: http://abjad.mbrsi.org/
  - LilyPond: http://lilypond.org/
- could also just run midis through Finale or Sibelius
- Since the output is a discrete set of (88) notes, cost function can be defined as the percent error
- Piano keys are numbered (lowest is 0, highest is 87), so could also look at how far off the output is from the true note
- Recurrent neural nets seem like best option since they handle sequences of values well

Workplan

- Try to use the masters thesis' neural net as starting point, and modify for .wav input
- First step is to get data processed into appropriate format. With MAPS all that would be needed is to run .wav files through python's fft algorithm
- Could then spend rest of time tweaking the already made neural net to see what gives the best results, or attempt building a net from scratch