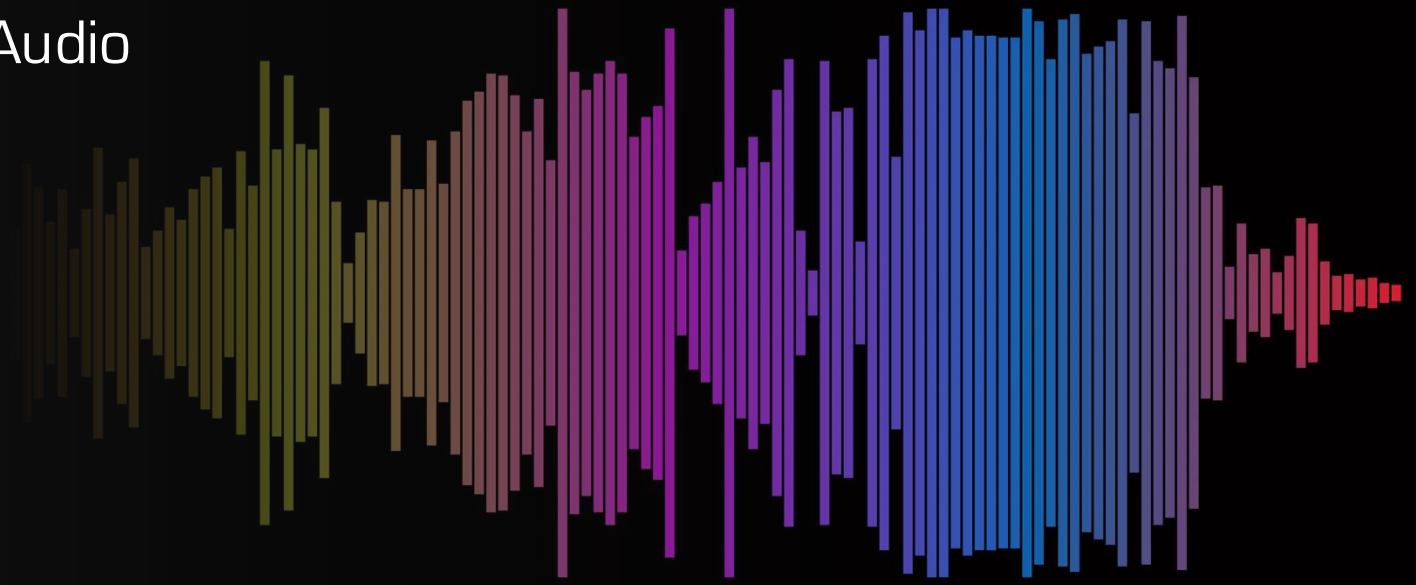
FXNet

Convolutional Neural Network for Audio Effects Classification

Christopher Relyea

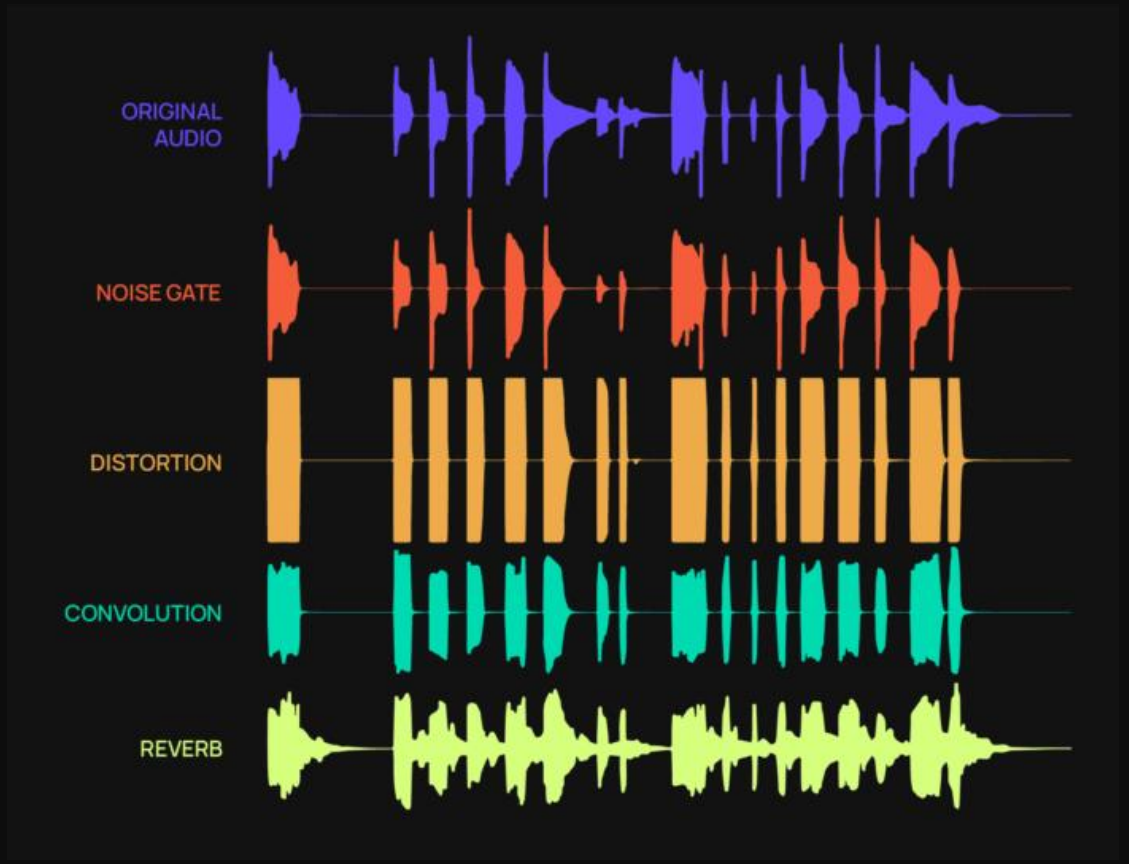
DS 340

Final Project



Background

- Audio effects transform a “dry” signal
- Four primary examples are Reverb, Distortion, Chorus, and Delay
- **Problem Statement**
  - I want to design a model that can recognize which effects have been applied to an audio signal and assign labels
- Why?
  - Music classification
  - Could be a useful tool for producers
  - Apply existing knowledge to this class



Audio Effects

Source

Reverb

“Wide” or “roomy” sound

Chorus

”Shaky” or “warbly” effect

Distortion

Crunchy and low-quality

Delay

Echo effect

Audio Effects (cont.)

Reverb + Delay + Chorus

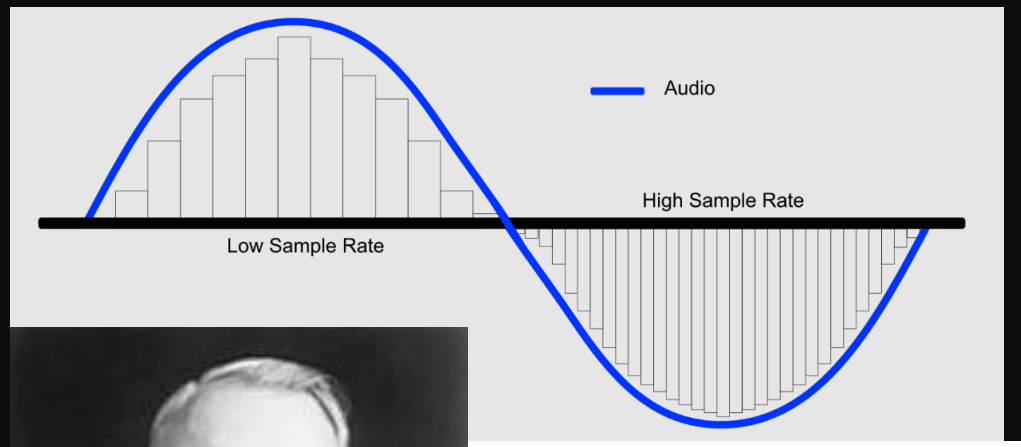
Distortion + Chorus

Challenge: Data Collection

- Source data from BandLab Sounds
  - 40 of each dry instrument guitar, keys, bass, voice
  - 96 “misc”
- Need to find a way to render each with effects (DAW)
  - Would take quite a long time by hand
- Found a DAW, Reaper, which supports Python scripting
- Rendered each audio file with all 16 combinations of distortion, chorus, delay, reverb in that order
- Effects plugins
- Total of 4112 audio samples to train with

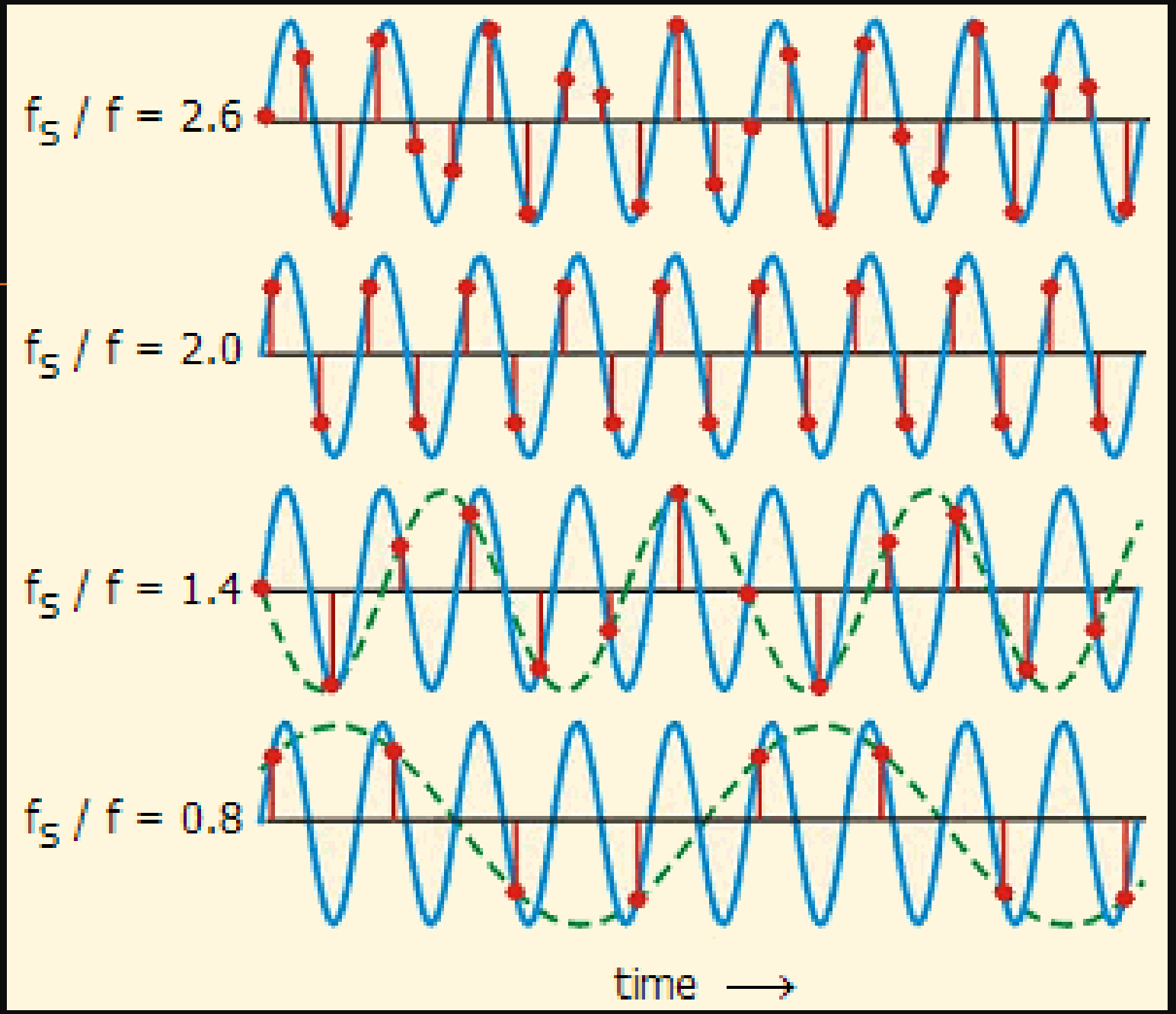
Methods (Audio Representations) and Fun Fact

- WAV files – lossless audio
  - A bit too detailed
- Sample rate
  - 44.1 kHz? 22.05 kHz? What does it mean?
  - **Fun Fact** - Nyquist Theorem, video recording, the origins of digital audio, 80 minutes on a CD
- Stereo vs. mono
- Duration and padding



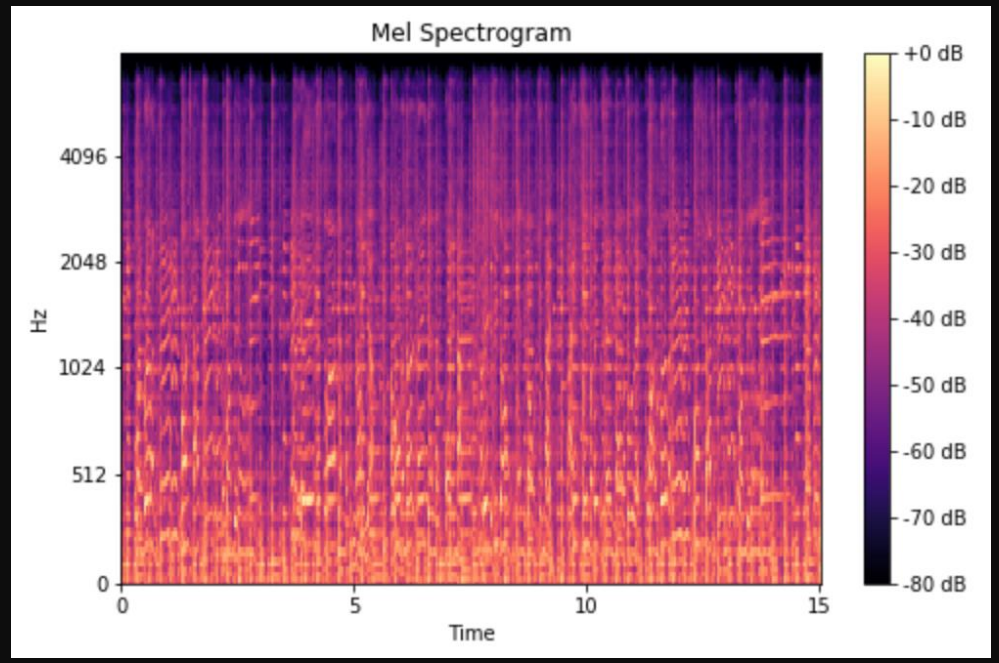
Why 44.1?

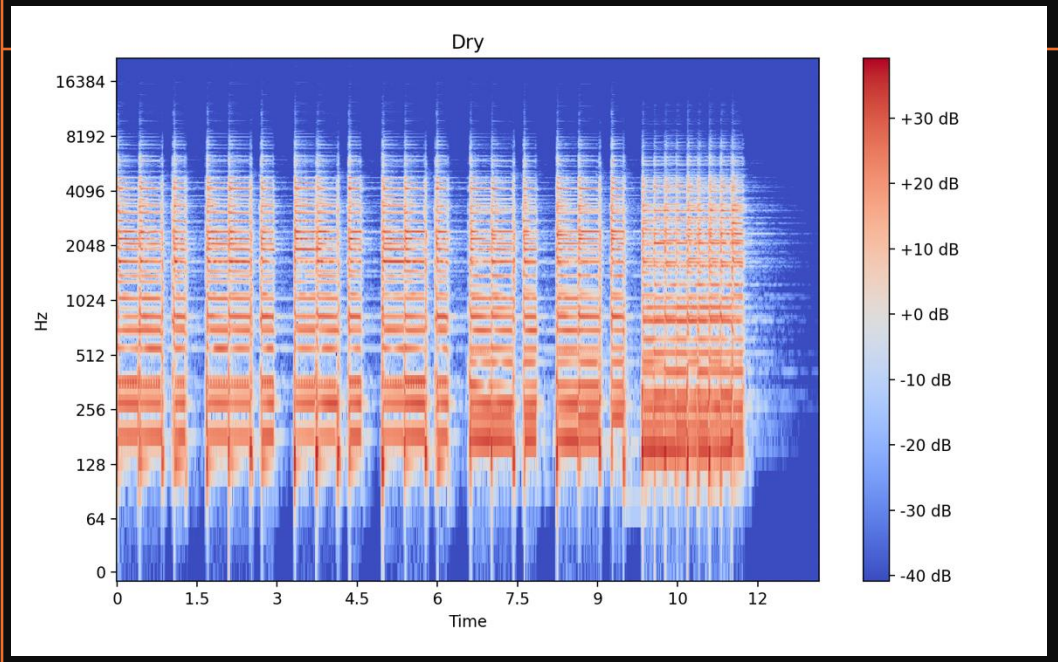
2 * 2 * 3 * 3 * 5 * 5 * 7 * 7 = 44100

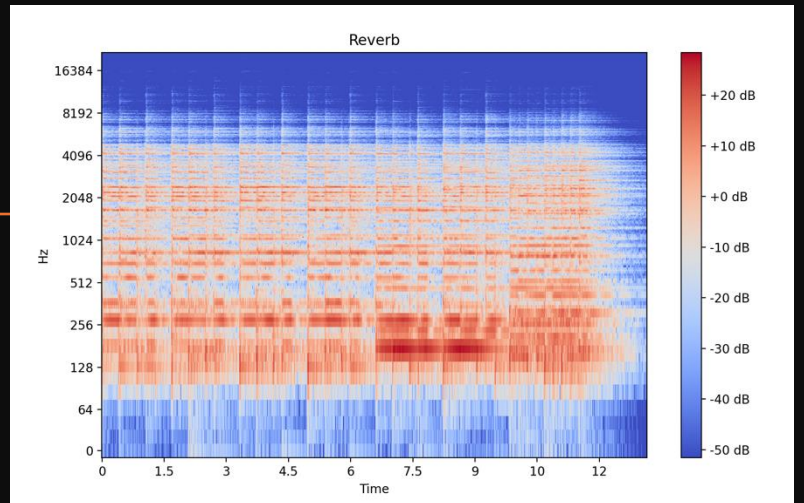

f_S / f = 2.6
f_S / f = 2.0
f_S / f = 1.4
f_S / f = 0.8
time ⟶

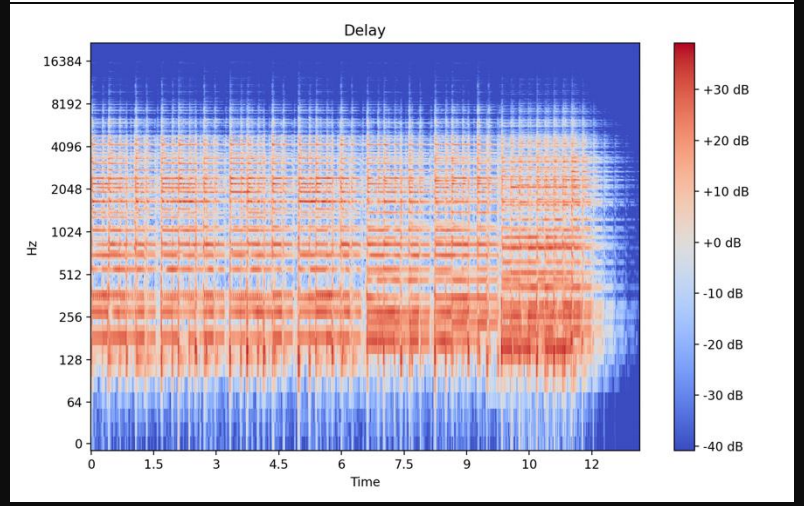
Methods: Mel Spectrograms

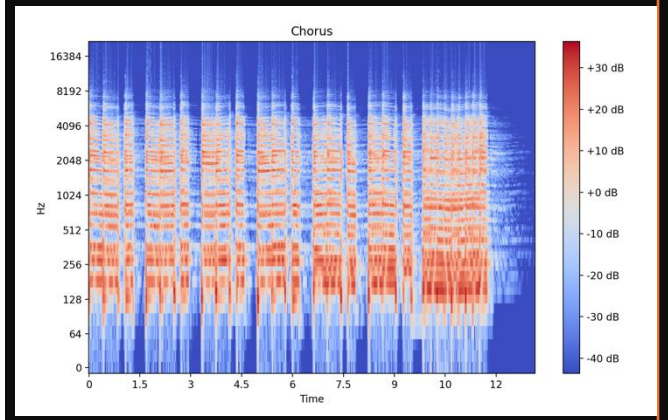
- Commonly used format for audio ML models
- Captures frequency change over time
- Like an image of the audio
  - Analogy: the colors of an image

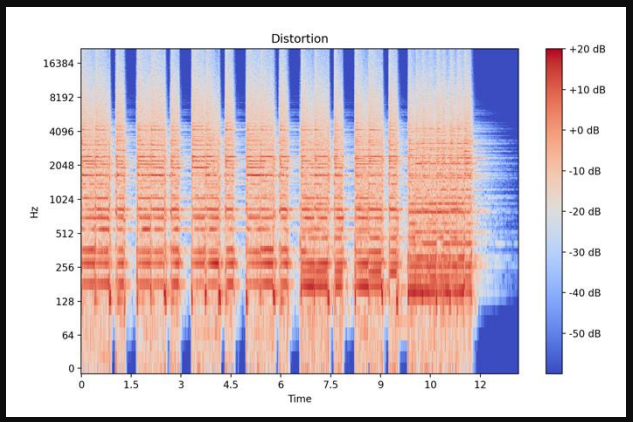



Dry
16384
8192
4096
2048
1024
512
256
128
64
0
Hz
0
1.5
3
4.5
6
7.5
9
10
12
Time
+30 dB
+20 dB
+10 dB
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB


Reverb
16384
8192
4096
2048
1024
512
256
128
64
0
Hz
0
1.5
3
4.5
6
7.5
9
10
12
Time
+20 dB
+10 dB
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB

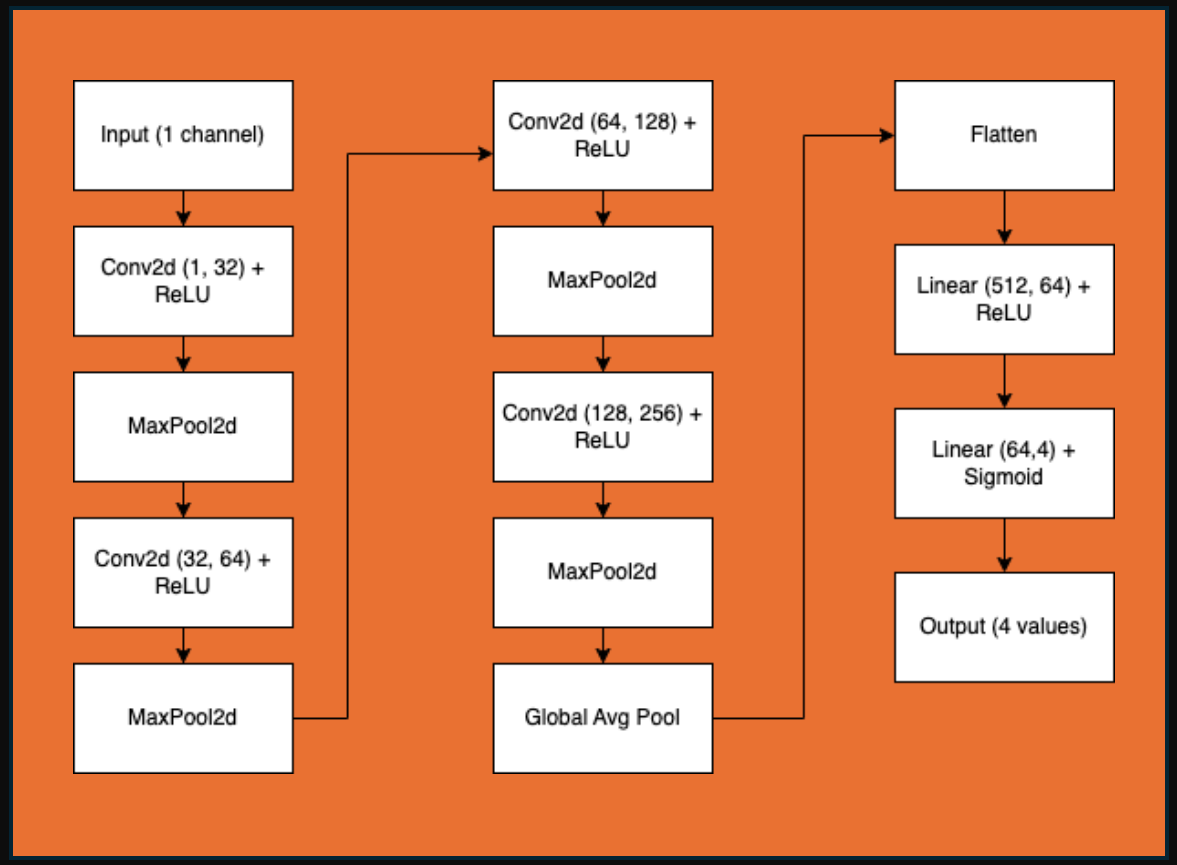

Delay
16384
8192
4096
2048
1024
512
256
128
64
0
Hz
0
1.5
3
4.5
6
7.5
9
10
12
Time
+30 dB
+20 dB
+10 dB
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB


Chorus
16384
8192
4096
2048
1024
512
256
128
64
0
Hz
0
1.5
3
4.5
6
7.5
9
10
12
Time
+30 dB
+20 dB
+10 dB
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB


Distortion
16384
8192
4096
2048
1024
512
256
128
64
0
Hz
0
1.5
3
4.5
6
7.5
9
10
12
Time
+20 dB
+10 dB
+0 dB
-10 dB
-20 dB
-30 dB
-40 dB
-50 dB

Methods: Creating and Adjusting the Network

- Started with a 4-layer CNN based on class examples, PyTorch documentation, and Keras examples
- Adapted from image classifier and audio classifiers
- Tools: PyTorch, Librosa, Colab
- Audio parameters:
  - Mel spectrogram bin size
  - Duration
  - Number of channels
- Hyperparameters
  - Early stopping
  - Learning rate
  - Batch size
- Data augmentation
  - Time shifting



Results

```
Per-label Metrics:
Distortion: Precision: 0.9532, Recall: 0.8953, F1: 0.9233
Chorus: Precision: 0.5200, Recall: 0.7792, F1: 0.6237
Delay: Precision: 0.6055, Recall: 0.8987, F1: 0.7236
Reverb: Precision: 0.8778, Recall: 0.9841, F1: 0.9279
```

- This is about the best the model can get
- Reverb and Distortion learned easily, which makes sense (why?)
- Delay is not great, chorus is worse
- Best model: 5 layers, dropout, mel bins = 256, batch size = 32, SR = 22.05 kHz, stereo audio, 75 epochs
- Better to use F1 than accuracy metric here. Why?

Conclusions

- It does not take much to recognize reverb and distortion, but delay and chorus are more subtle
- Future work
  - Consider real music production examples more – background noise and other instruments
  - More augmentations
  - Other representations – raw WAVs?
  - Try to build a classifier for just chorus and delay
  - More effects