AI Solutions: Audio to MIDI

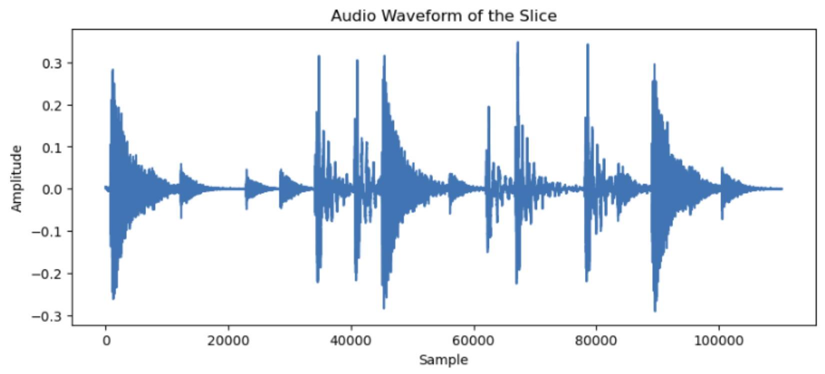
ROBERT McKEAN
GLOBAL ACTIVE, LLC

Table of Contents

Project Goal

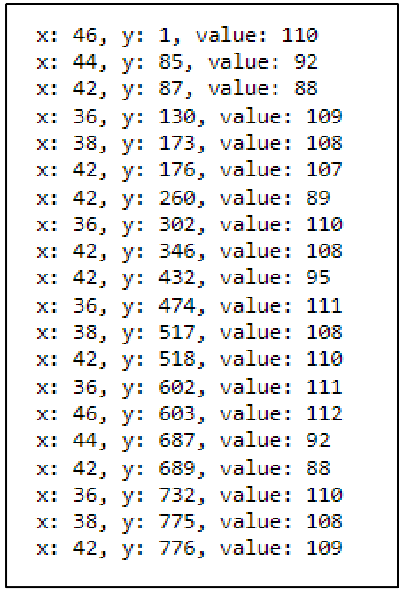
Create a MIDI file from any random audio recording of an isolated drum track.

Input: drum recording (wav)



Convolution-based AI modeling

Output: midi file



```
x: 46, y: 1, value: 110
x: 44, y: 85, value: 92
x: 42, y: 87, value: 88
x: 36, y: 130, value: 109
x: 38, y: 173, value: 108
x: 42, y: 176, value: 107
x: 42, y: 260, value: 89
x: 36, y: 302, value: 110
x: 42, y: 346, value: 108
x: 42, y: 432, value: 95
x: 36, y: 474, value: 111
x: 38, y: 517, value: 108
x: 42, y: 518, value: 110
x: 36, y: 602, value: 111
x: 46, y: 603, value: 112
x: 44, y: 687, value: 92
x: 42, y: 689, value: 88
x: 36, y: 732, value: 110
x: 38, y: 775, value: 108
x: 42, y: 776, value: 109
```

The Output midi file contains the midi note number (x), relative timing (y, 3ms resolution), and note velocity (value).

Steps to Create and Verify Data and Model

1. Use a digital audio workstation (daw) to combine thousands of midi drum loops into a single midi file.
2. Using multiple drum kits with different sounds, record a mono audio file from the midi data.
3. Note: For this project, the file length was 9.5 hours.
4. Create 2.5 second indexed ‘slices’ of the audio and midi. Ensure the slice indexes remain time aligned.
5. X data: wav values of the audio slice. Inside the model, convert to a short-time Fourier transform (STFT).
6. Y data (training): Create heat map of the know midi values for the correlated audio slice.
7. So now we know the input (audio data) and expected output (heat map of midi data). We can train the model.
8. Train a convolution-based model to generate a new midi heat map based upon the audio input.
9. To visually confirm results, compare the known midi slice heat maps to the generated midi slice heat map.
10. From the new heat map, extract the note value, relative timing, and velocity. Select threshold to reduce noise.
11. Generate a new midi file from the midi heat map and import into the daw. Play the midi file using drum sounds.
12. Can also compare original midi slice to generated midi slice by printing midi data.

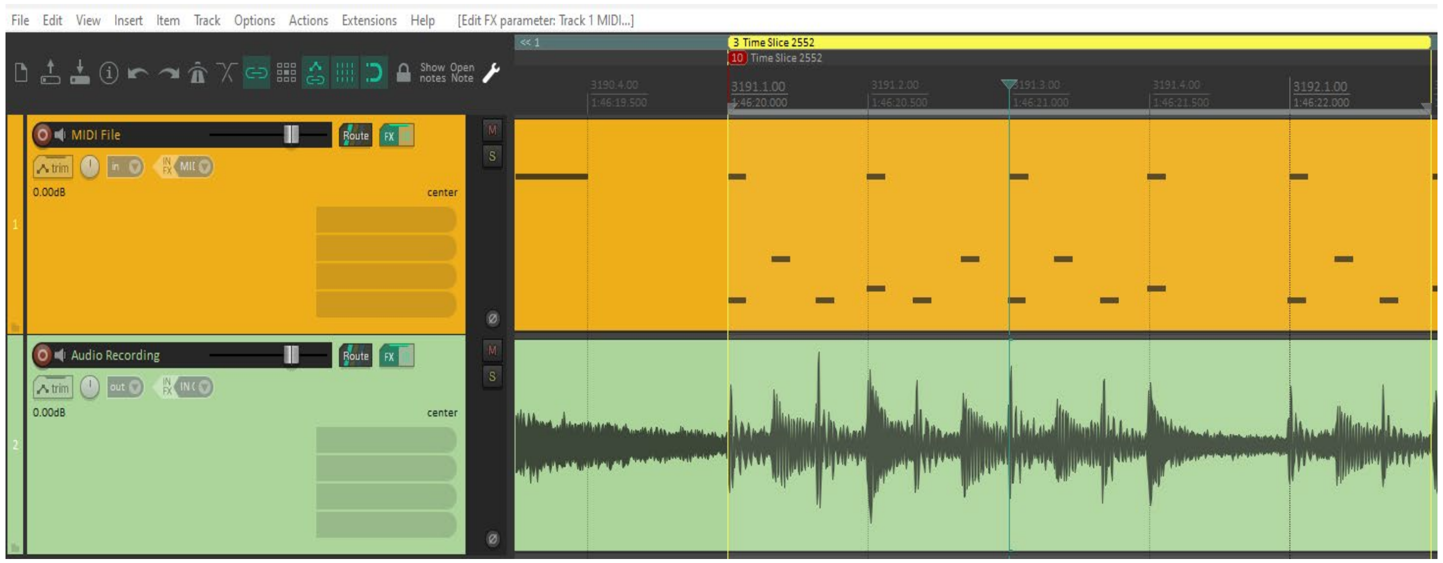
Create the MIDI and Audio Files

The MIDI file combines thousands of MIDI loops into one long file (9.5 hours).

Audio files were then created by using a drum plug-in.

Multiple drum kits were used to provide a variety of sounds for training.

DAW software showing MIDI file and generated drum audio.



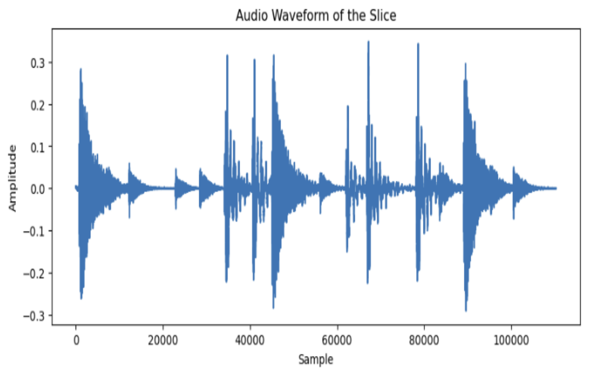
Drum plug-in used to generate audio file.

Training the Model

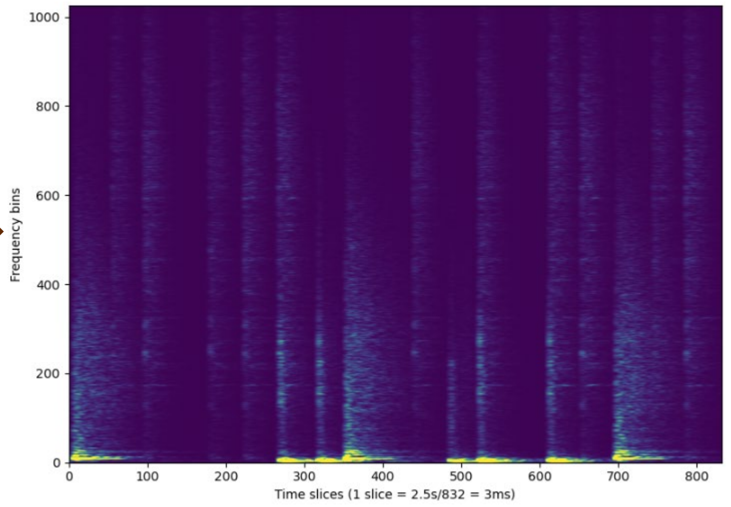
Note that the STFT data aligns with the note data in the target output.

Input: drum recording (wav)

torch.Size([1, 110250])

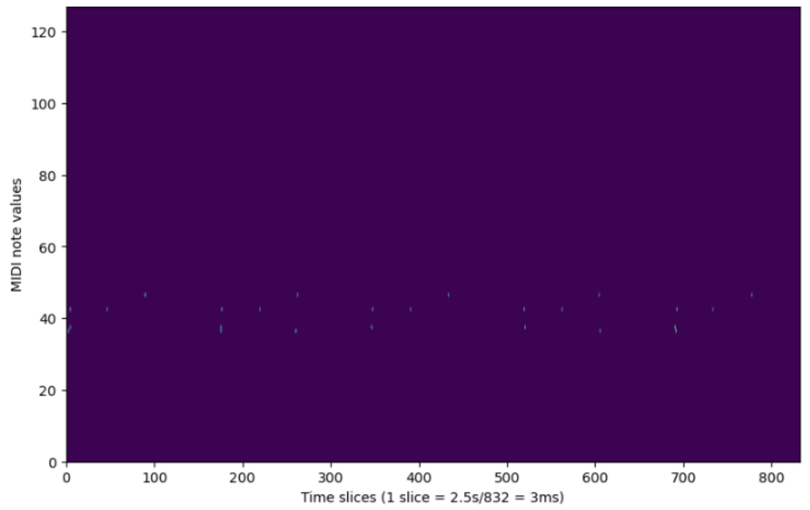


STFT of Input (nnAudio)



C O N V O L U T I O N

Target Output (Y) – MIDI data



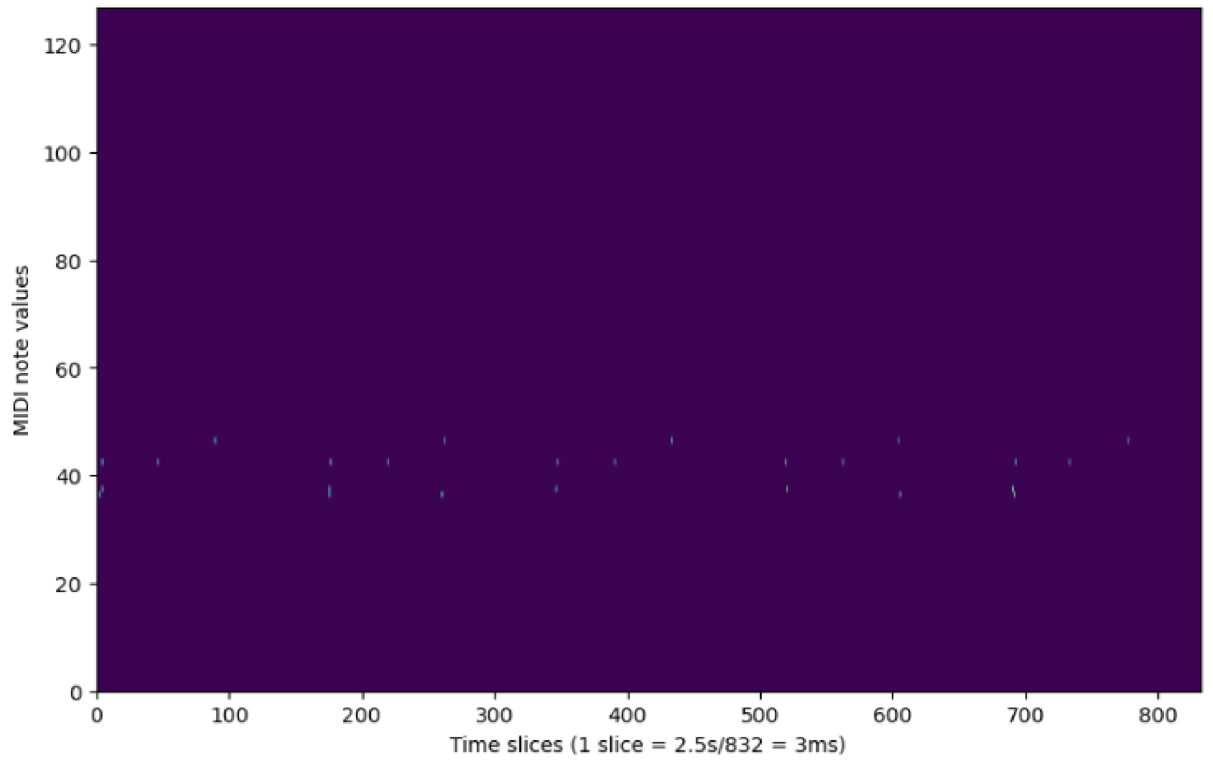
X data for training

Model

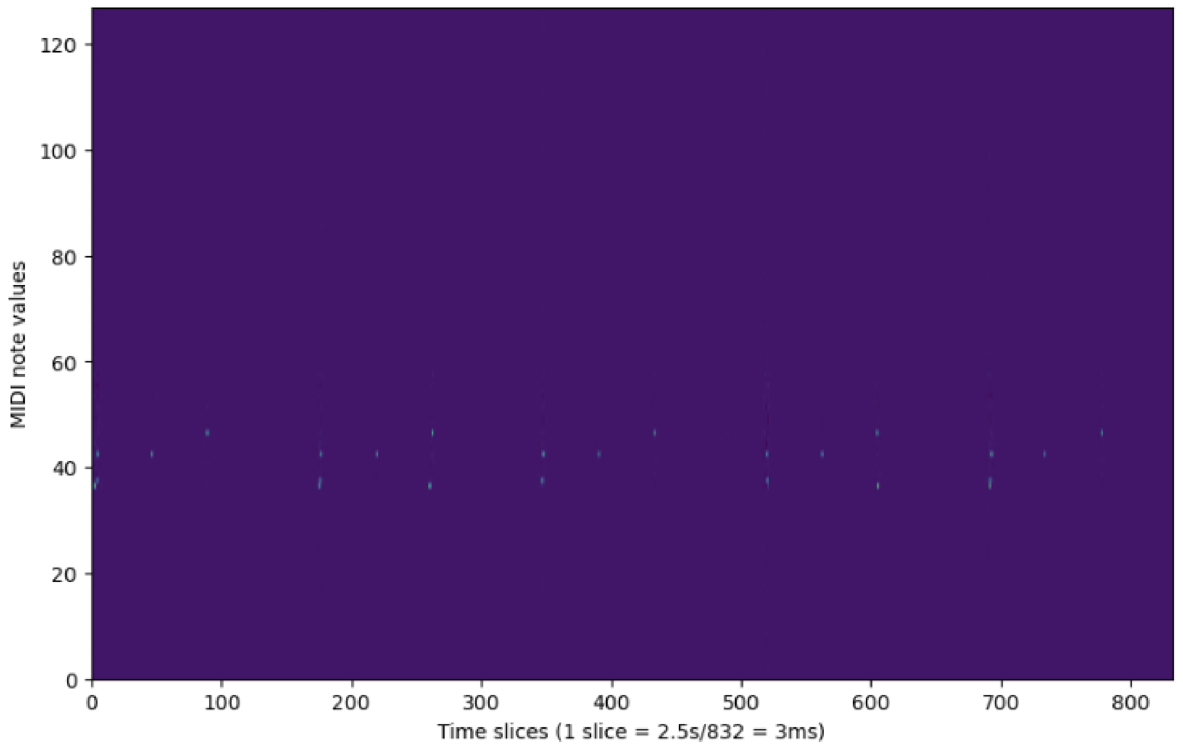
Y data for training

Results – Target Output vs Actual Output

Target Output (Y) – MIDI data



Actual Model Output – MIDI data



The results are very similar. At times 170 and 680, there is minor smearing of near-simultaneous drum hits.

Results – Actual Values

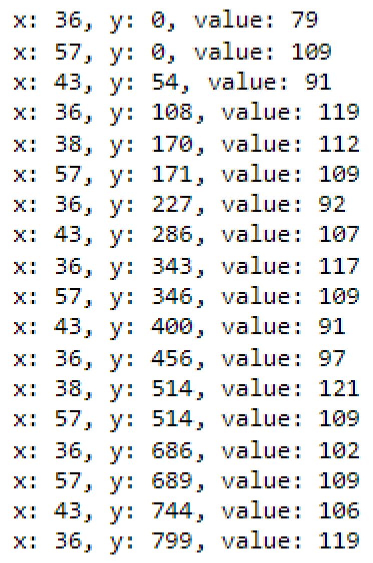
For this slice (2552), the note (x) and relative time (y) values match exactly.

The values (note velocity) are higher on the post-modeled files. More detailed scaling analysis is required.

For this example:

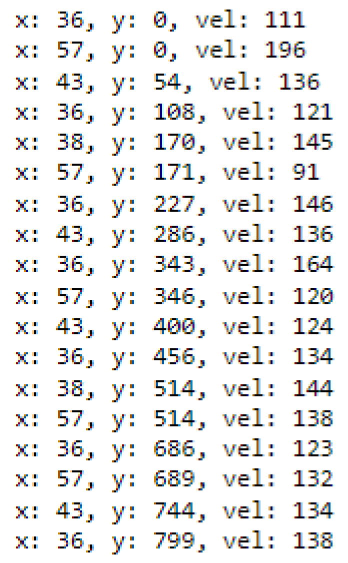
1. Slice number: 2552
2. Threshold: 0.5

Original midi file



```
x: 36, y: 0, value: 79
x: 57, y: 0, value: 109
x: 43, y: 54, value: 91
x: 36, y: 108, value: 119
x: 38, y: 170, value: 112
x: 57, y: 171, value: 109
x: 36, y: 227, value: 92
x: 43, y: 286, value: 107
x: 36, y: 343, value: 117
x: 57, y: 346, value: 109
x: 43, y: 400, value: 91
x: 36, y: 456, value: 97
x: 38, y: 514, value: 121
x: 57, y: 514, value: 109
x: 36, y: 686, value: 102
x: 57, y: 689, value: 109
x: 43, y: 744, value: 106
x: 36, y: 799, value: 119
```

Post-modeled midi file



```
x: 36, y: 0, vel: 111
x: 57, y: 0, vel: 196
x: 43, y: 54, vel: 136
x: 36, y: 108, vel: 121
x: 38, y: 170, vel: 145
x: 57, y: 171, vel: 91
x: 36, y: 227, vel: 146
x: 43, y: 286, vel: 136
x: 36, y: 343, vel: 164
x: 57, y: 346, vel: 120
x: 43, y: 400, vel: 124
x: 36, y: 456, vel: 134
x: 38, y: 514, vel: 144
x: 57, y: 514, vel: 138
x: 36, y: 686, vel: 123
x: 57, y: 689, vel: 132
x: 43, y: 744, vel: 134
x: 36, y: 799, vel: 138
```

Project Notes and Lessons Learned

1. Results were very good for drums used for training. To make results more generic, more curated drum samples are needed.
2. There was some overfitting due to the limited set of curated drum samples available.
3. More tuning of the model would be helpful, including steps that would increase GPU memory consumption.
4. GPU memory consumed: 2.3 GB.
5. System memory consumed: 20 GB.
6. MIDI note resolution: 3ms. Could be decreased at the expense of more GPU memory and longer training.
7. Tip: Input tensor length must be compatible with convolution algorithms. This program needed to truncate input tensor from 862 to 832 samples.
8. Tip: Create a test tensor with random data to confirm that the model produces the expected output data format. The output from the model must match the format of the Y data.

In summary, I was able to:

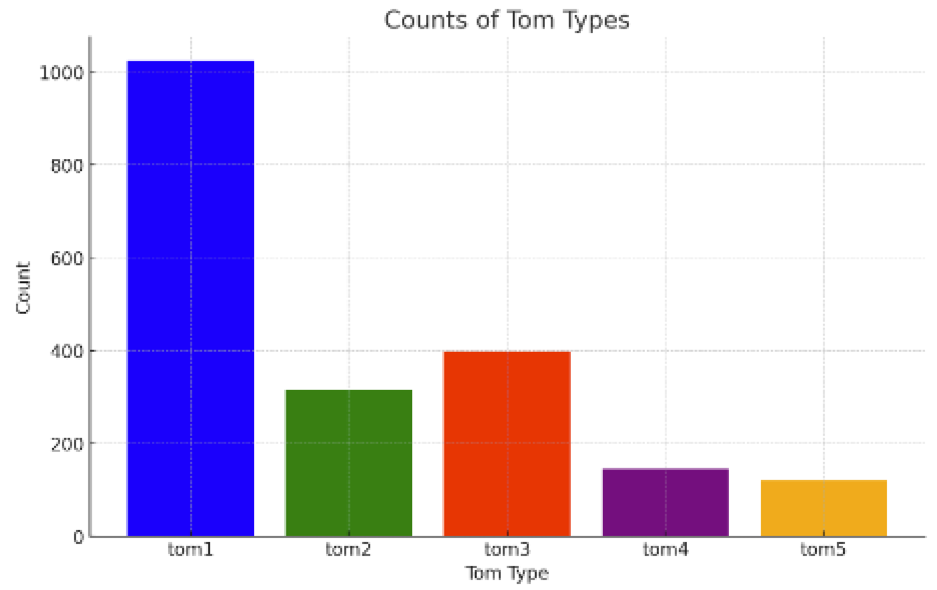
1. Identify a unique solution to a problem.
2. Develop a strategy for generating X/Y data that could be used for modeling.
3. Create large synchronized audio and MIDI files.
4. Manipulate/fix data gaps in Excel (Data Science).
5. Create and execute a convolution model from scratch to achieve excellent results (AI Development).
6. Developed k-means clustering to categorize drum types.
7. Use Python and PyTorch to deliver a working solution.

Biggest Challenges

1. Quality drum data. Used drum samples from Kontakt, EZ Drummer 3, and individual one-shots.
2. Even with a folder full of individual drums (toms, for example), there will be outliers. Electronic drums are different sound, for example.
3. Needed: A deep set of curated data. With 10,000's of drums, curating this data was the largest challenge.
4. Developed unsupervised K-clustering to sort and classify drums. Effective.
5. Memory. Deeper algorithms use more memory and take longer to model. Needed to keep GPU memory below 8GB to run on laptop's NVIDIA GeForce RTX3080.

Below are the results from K-clustering approximately 2,000 individual tom tom drum samples.

The primary separator was the tone, with tom1 being the lowest in frequency.



Thank you!