Sound to Symphony

AI-Powered Generation for Musical Creativity

Nathan H. Arnett, Philip d'Arenberg, Christelle Joseph, Myoungchul Kim

"Life without music would be a mistake"

Friedrich Nietzsche

App

Our app generates original musical composition from deep learning.

Made for Live Performance
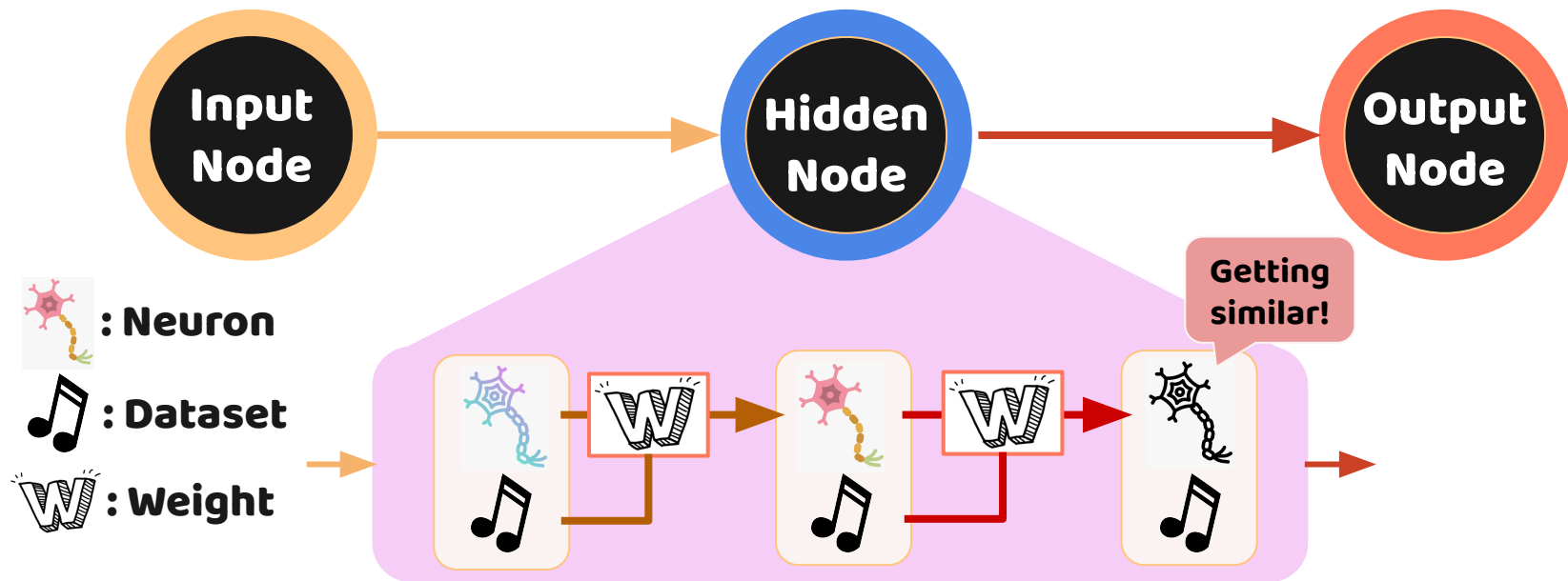

For Musicians and Creatives

API Integration

Recurrent Neural Networks

Input Node → Hidden Node → Output Node

Getting similar!

: Neuron

: Dataset
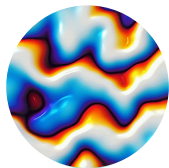
: Weight

Neuron learns the pattern of compositions from dataset

Dataset

MAESTRO

- 200 hours of piano recordings
- All classical music
- MIDI files
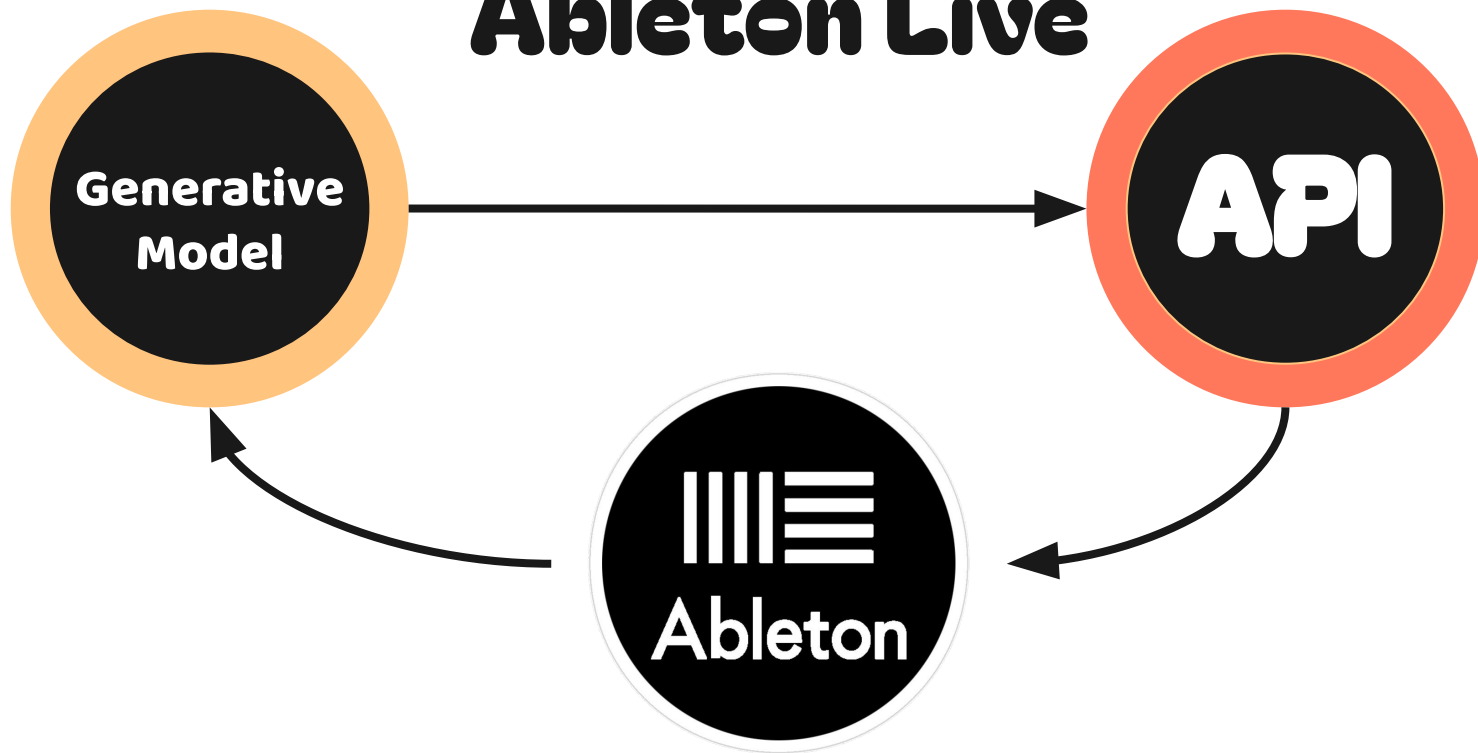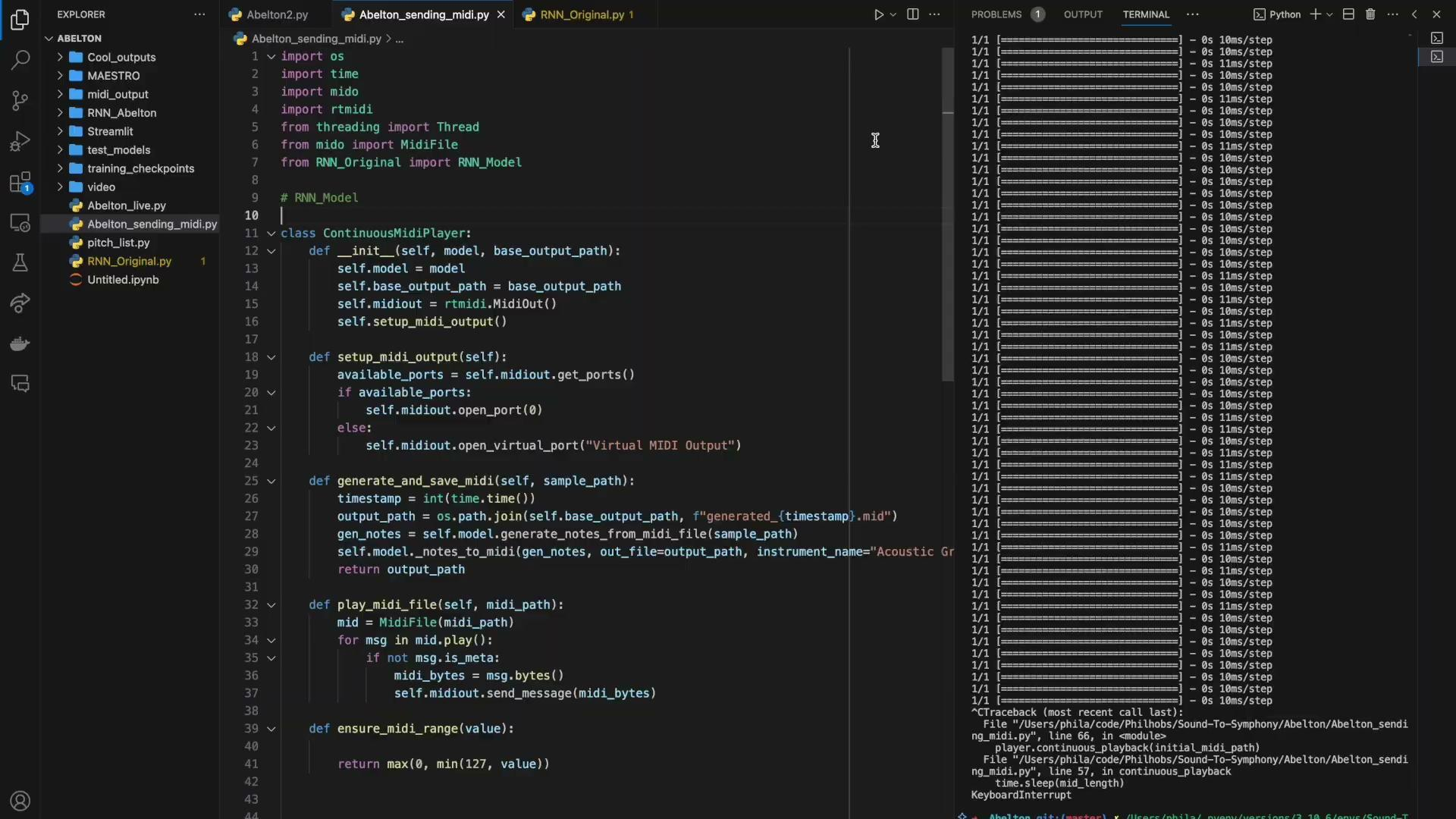
MIDI

- Do not contain sound information
- Contains pitch, velocity and duration
- Instructions to musical software

User Interface

https://sound-to-symphony-hggkk6
gcyupalttasetpiw.streamlit.app/

Ableton Live

```python
import os
import time
import mido
import rtmidi
from threading import Thread
from mido import MidiFile
from RNN_Original import RNN_Model

# RNN_Model

class ContinuousMidiPlayer:
    def __init__(self, model, base_output_path):
        self.model = model
        self.base_output_path = base_output_path
        self.midiout = rtmidi.MidiOut()
        self.setup_midi_output()

    def setup_midi_output(self):
        available_ports = self.midiout.get_ports()
        if available_ports:
            self.midiout.open_port(0)
        else:
            self.midiout.open_virtual_port("Virtual MIDI Output")

    def generate_and_save_midi(self, sample_path):
        timestamp = int(time.time())
        output_path = os.path.join(self.base_output_path, f"generated_{timestamp}.mid")
        gen_notes = self.model.generate_notes_from_midi_file(sample_path)
        self.model._notes_to_midi(gen_notes, out_file=output_path, instrument_name="Acoustic Gr
        return output_path

    def play_midi_file(self, midi_path):
        mid = MidiFile(midi_path)
        for msg in mid.play():
            if not msg.is_meta:
                midi_bytes = msg.bytes()
                self.midiout.send_message(midi_bytes)

    def ensure_midi_range(value):

        return max(0, min(127, value))
```

Terminal output:
```
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 11ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
1/1 [==============================] - 0s 10ms/step
^CTraceback (most recent call last):
  File "/Users/phila/code/Philhobs/Sound-To-Symphony/Abelton/Abelton_sendi
ng_midi.py", line 66, in <module>
    player.continuous_playback(initial_midi_path)
  File "/Users/phila/code/Philhobs/Sound-To-Symphony/Abelton/Abelton_sendi
ng_midi.py", line 57, in continuous_playback
    time.sleep(mid_length)
KeyboardInterrupt
```

TAKE AWAY

GENERATIVE MUSIC COMPOSITIONS

CUSTOMIZABLE MIDI OUTPUTS

LIVE MUSIC INTEGRATION

"You are the music while the music lasts"

T.S. Eliot

THANK YOU !

Myoungchul Kim, Christelle Joseph, Philip d'Arenberg, Nathan H. Arnett

Table of contents