Music Generation Team Project Report

Arseny Savchenko, Amir Bikineyev, Louay Farah, Timur Zheksimbaev

Innopolis University

25 July, 2023

I Introduction

This report aimed to provide an overview of the final project, analyze strong points and weaknesses and what can be further improved in the current state of the development. The goal of the project is to generate MIDI files with multiple batches.

II Algorithms

Resulting music contains 4 batches: lead melody, arpeggio, harmony and drums. Every batch generation is assigned to a specific person (lead melody - Savchenko, Harmony - Bikineyev, arpeggio - Farah, drums - Zheksimbaev). User is free to manually choose any key, scale or mode with an input in the beginning of the program. On every step, the resulting .mid file is passed further and combined with the next batch.

A. *Lead Melody Generation*

First bar is generated with a strict random algorithm: it traverses through all 16 parts of the bar and puts notes or creates the delay. If note's position % 4 == '0', note's length can be either 1, 2 or 4; '1' - length is 1; '2' - length is 1 or 2; otherwise - 1.

Algorithm relates to a chosen melody type (ABAB, AAAB or ABAC). 'A' is always generated with the algorithm above. For ABAB, 'B' is generated as the mutation of 'A' (some notes may change their pitch). For AAAB, 'B' is generated with the algorithm above. For ABAC, 'B' is new, 'C' is the mutation of 'A'.

Result is checked with the filters. Finally, all generated 4 bars are duplicated to generate 8 bar melody and settings file with chosen parameters is generated.

B. *Harmony Generation*

Variant of evolutionary strategy was used to generated chords. Initially, population consists of 100 members constantly. Mutation is done as changing the chord type among the possible in the key of melody. Fitness function contains the pleasant parameter that checks whether the note from melody and chords are matched; the chord progression parameter that checks whether chords lies in some progression and the repetitions of the same chord in a row parameter to avoid the same chord 3+ time in a row.

C. Arpeggio

Algorithm generates arpeggios for an input melody by first analyzing the key of the melody using the analyze() method of the music21 library, and then looping through the measures in the melody. For each measure, it creates a Chord object from the notes in the measure and transposes it down by 2 octaves. Then applies a custom arpeggiate function to the Chord object to generate an arpeggio, where the style and direction of the arpeggio are chosen randomly from a set of ”up” and ”down”, ”ascending” and ”descending”. The arpeggio is added to an arpeggio stream, which is then combined with the original melody to create a new score.

D. Drums Generation

Two drum patterns and three bass patterns were manually composed. Then, two drum patterns and bass pattern were randomly combined. Drum patterns consist of kicks, snares and hi-hats.

III Analysis of the result. Further improvements

Despite being universal and unique on every try, melody generation requires multiple tries to generate something catchy. As for the chords, additional mutations and crossover can be used. To improve the arpeggios, More variation to the arpeggio patterns can be considered by introducing additional patterns and by allowing for more randomness in the pattern selection. As for the drums, more complex drum and bass patterns, with usage of many other instruments (cymbals, toms, crashes, etc.).