Polyphonic Reduction for Playable Piano Arrangement

Luke Cheng, Sanil Desai, Michael Lu, Thomas Lin

CSCI 1470 Fall 2025 Final Project

Introduction

As digital music production becomes increasingly widespread, MIDI files have become the norm for representing virtually any form of musical content. However, these files are designed for digital playback rather than for human pianists. Our project aims to enhance the human feasibility of digital music transcription and MIDI files. We focus on polyphonic reduction: given a complex musical input sequence, produce a simplified two-hand piano arrangement that preserves the essential melodies, harmonies and character of the original. Our high-level approach follows a sequence-to-sequence framework, achieved through a transformer architecture. We create a self-supervised pipeline that corrupts clean arrangements with noise, and then trains the model to remove noise to recover the original arrangement.

Data

We use the **POP909** dataset, which contains approximately **2,000** MIDI files of Chinese pop music. Each file consists of multi-track arrangements that include melody, bridge and accompaniment tracks.

Results

Playability Index

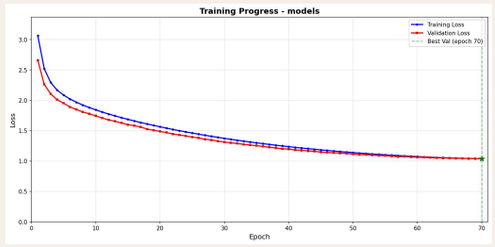
Total Unplayable Chords: the total number of unplayable chords across 20 outputs is 43 chords.
Unplayable Rate: the unplayable rate is 0.4 per file.

Musical Quality

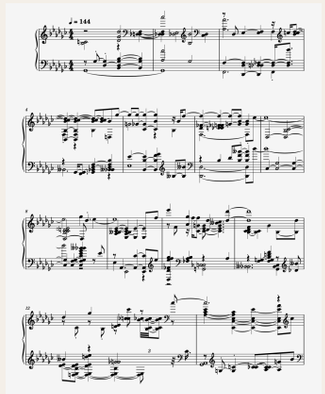
Average Pitch Similarity: 0.817 (1.0 = identical pitch distribution)
Average Density Ratio: 0.63 (1.0 = same note count as ground truth)

Image Segmentation

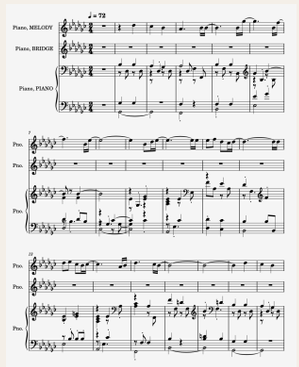
Average F1 Score: 0.312
Average Precision: 0.367
Average Recall: 0.297



Training & Validation Loss



Output



Input

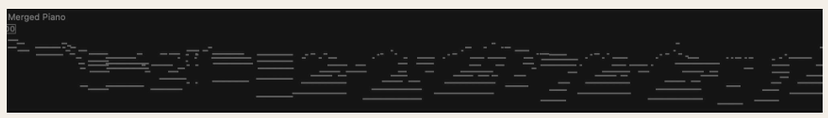
Methodology

1 Data Cleaning

We automatically identify left and right hand splits by unsupervised K-means clustering. All notes are merged into a single track, pitch features are extracted, and K=2 clustering separates high and low register notes, which we map to right and left hands respectively. This produces clean two-hand "ground-truth" labels for training.

2 Data Corruption

We corrupt the clean arrangements by adding noise (specifically shifted duplicate notes, octave doubling, nearby tones, passing tones and more) to create unplayable dense inputs. Subsequently, we transpose each arrangement into two additional random keys, improving generalization across pitch ranges and tripling our dataset size. We then split this augmented dataset into 90% training and 10% validation.



3 Tokenization

We convert the MIDI files into sequences of tokens following our simplified REMI tokenization scheme, specifically with 16th-note timing maximum resolution; reduced four dynamic levels; piano-only tokens; and implicitly-encoded rests. Moreover, we split long pieces into overlapping 512-token chunks, with 50-token minimum length to ensure adequate context.

4 Transformer Model

We train a T5-style transformer for sequence-to-sequence reduction. The encoder contains 6 layers of attention to process corrupted inputs, with 6 decoder layers (each with 8-head multi-attention) that attend to encoder hidden states to generate simplified token sequences. Afterwards, we enforce playability by adjusting every chord that spans over 12 semitones. We reduce these to only the outer interval, with up to two inner notes within one octave.

Discussion

Generally, the transformer produces playable arrangements, with a notably low unplayability rate of 0.4 per MIDI. We find our output to generally preserve the the melody and musical elements of the original, while somewhat simplifying the complicated unplayable sections. However, the transformer does not have a true or accurate fundamental understanding of musical composition. Notably, the musical ties, abundance of accidentals, misplaced rests and highly note-dense chords are uncharacteristic of typical sheet music, although some of this may be attributed to the default MIDI-to-sheet music conversion.

On the whole, however, the model exhibits particularly low F1 score, precision and recall, all of which are below 0.4. This suggests that the model is still struggling to distinguish between noise and important notes that should be kept in the arrangement. As such, the model is creating arrangements that deviate from the clean, "ground-truth" arrangements, even though the model generally preserves playability and melody during audio playback.

We learned that the model's performance is majorly dependent on total training time. We also learned that optimizing a loss function could potentially be much more efficient and effective for these smaller models. For example, upon reflection we consider a proposed loss function: $L_{total} = L_{CE} + \lambda_1 \cdot L_{span} + \lambda_2 \cdot L_{voice} + \lambda_3 \cdot L_{ergonomic}$ where L_{span} penalizes chords exceeding playable spans, L_{voice} encourages smooth voice leading, and $L_{ergonomic}$ rewards natural hand positions. The issue with this function is that many established playability constraints are not differentiable (step-wise), and computational costs of calculating this loss function is much higher, which would further slow the training process.

Challenges

1. **Playability and Loss:** Our goal to optimize for playability is difficult as at its core it is a binary, non-differentiable metric. The approach we took was to simply optimize for simplicity by adding unplayable noise and training the model to remove the noise while distilling the melody. However, other possible solutions include engineering a loss function that includes metric such as range, tempo, jumps, etc.
2. **Training Time:** Since these models in the pipeline are generative, training is highly unstable and very computationally intensive. Getting feedback from the model took an extensive time and made it such that iteration on architecture design was extremely unfeasible.
3. **Data scope:** POP909 is a very narrow-scope dataset when compared to all music. Ideally, we would have a much larger dataset to better sample For example, POP909 primarily comprises Chinese pop songs, which all tend to have similar musical characteristics, thus limiting the scope and generalizability of the model. Increasing dataset diversity would also require more time and compute than we had given our constraints.
4. **Chunk Based Processing:** Long songs are divided into 512 token chunks that are processed differently and then recombined. This can lead to odd end tokens and other discontinuities. A possible solution would to be train with overlapping windows, however this is infeasable given our resources due to memory limitations.