

APPENDICES

This section aims to offer a comprehensive explanation of the technical concepts and mathematical formulations discussed in the paper, along with additional examples and case studies to better illustrate their practical applications.

A DETAILS FOR ENCODER, DECODER AND TRAINING

A.1 E-MLP AND D-MLP STRUCTURE

We denote a five-attribute note as $N = [O, D, P, V, I]$, where O represents the onset time, D stands for duration, P denotes pitch, V refers to velocity, and I indicates instrument. A vocabulary is constructed to encompass all possible tokens corresponding to these five attributes.

E-MLP (Encoder MLP). The E-MLP model is responsible for encoding the input tokens of the note. It consists of a token embedding layer that maps each input token to a dense vector representation. Specifically, each token is passed through an embedding layer, followed by a series of linear transformations and activation functions. The output is a 16-dimensional vector for each note. Mathematically, the process can be represented as:

$$E_{\text{MLP}}(N) = \sigma(W_2(\sigma(W_1 T_N + b_1)) + b_2)$$

where: T_N represents the input token embedding for note N , W_1 and W_2 are the weight matrices for the two linear layers, b_1 and b_2 are the bias terms, and σ denotes the activation function ReLU. The final output of the E-MLP is a fixed 16-dimensional latent representation of the note N .

D-MLP (Decoder MLP). The D-MLP model aims to reconstruct or predict the five discrete attributes (O, D, P, V, I) from the latent representation generated by the E-MLP. The D-MLP consists of two linear-activation layers that are stacked together. For each attribute, a separate classifier is used, as each attribute takes discrete values from predefined categories. The process for predicting each attribute can be described as follows:

$$\hat{A}_i = \text{softmax}(W_i \cdot \sigma(W_{i-1} \cdot Z + b_{i-1}) + b_i)$$

where: Z is the latent 16-dimensional vector obtained from the E-MLP, W_{i-1} and W_i are weight matrices, with W_{i-1} belonging to the first linear layer and W_i to the classifier for attribute A_i (where $i \in \{O, D, P, V, I\}$), b_{i-1} and b_i are the bias terms, and softmax is applied to ensure that the output is a valid probability distribution over the possible discrete values of each attribute. Thus, the D-MLP outputs five probability distributions, one for each attribute of the note.

Overall Process. In summary, the E-MLP encodes the note into a fixed-length latent representation, while the D-MLP decodes this representation into the predicted values of the five attributes. The entire process can be formulated as:

$$\hat{N} = D_{\text{MLP}}(E_{\text{MLP}}(N))$$

where \hat{N} represents the predicted note attributes.

A.2 FORMULATION OF THE NOTE ENCODER

As stated in the paper, the note encoder takes a sequence of embedding vectors as input and utilizes a 4-head, 6-layer Transformer architecture. Denote each note as N , and let the embedding of the note be processed by the E-MLP, denoted as E_{MLP} . We use the final hidden state of the Transformer as the input to two separate output layers. Each of these output layers consists of a linear transformation followed by a non-linear activation function. These layers output two distinct features: note features and musical features.

The input to the Transformer is a sequence of embedding vectors, denoted as $X = [E_{\text{MLP}}(N_1), E_{\text{MLP}}(N_2), \dots, E_{\text{MLP}}(N_T)]$, where T is the length of the sequence, and each $E_{\text{MLP}}(N_i)$

is the 16-dimensional embedding of note N_i obtained from the E-MLP. The Transformer first maps these to its feature space, and then processes this sequence through its multi-head self-attention mechanism and a series of feed-forward layers. The final hidden state H_T at time step T (corresponding to the last note in the sequence) is used as the input to the subsequent output layers.

The Transformer-based encoding process can be formulated as:

$$H_T = \text{Transformer}(X)$$

where H_T is the hidden state vector at the last layer and the final time step of the Transformer.

We then apply two separate output layers to H_T , one for extracting note features and another for extracting musical features. Each output layer consists of a linear transformation followed by a non-linear activation function. The outputs are defined as follows:

$$F_{\text{note}} = \sigma(W_{\text{note}}H_T + b_{\text{note}}), F_{\text{music}} = \sigma(W_{\text{music}}H_T + b_{\text{music}})$$

Thus, the final formulation can be expressed as:

$$F_{\text{note}}, F_{\text{music}} = \text{Output Layers}(\text{Transformer}(E_{\text{MLP}}(N_1), E_{\text{MLP}}(N_2), \dots, E_{\text{MLP}}(N_T)))$$

A.3 LOSS FOR THREE-STEP TRAINING OF FEATURE EXTRACTION

Step 1: E-MLP and D-MLP. Given a note $N = [O, D, P, V, I]$ representing its five attributes (onset, duration, pitch, velocity, instrument), the task is to embed this note into a 16-dimensional vector using the E-MLP. The D-MLP then reconstructs the five attributes from this 16-dimensional representation. The loss function used to minimize the reconstruction error across the five attributes is formulated as:

$$\mathcal{L} = \sum_{i=1}^5 \text{CrossEntropy}(A_i, \hat{A}_i)$$

where A_i are the true attributes and \hat{A}_i are the predicted attributes. Each attribute is treated as a classification task using cross-entropy loss.

Step 2: Note Encoder and Bar Decoder. The note encoder processes a sequence of 16-dimensional embeddings, generating both note-level and musical-level features. The bar decoder takes the note features as input and auto-regressively generates the embedding vector at each position.

The loss function for the note embeddings is the Mean Squared Error (MSE) between the predicted and target sequences, both of which consist of 16-dimensional embedding vectors. This can be formulated as:

$$\mathcal{L}_{\text{note}} = \frac{1}{T} \sum_{t=1}^T \text{MSE}(E_t, \hat{E}_t)$$

where E_t and \hat{E}_t represent the true and predicted 16-dimensional embedding vectors at time step t , and T is the length of the sequence.

For the musical attributes (chord, time signature, tonality and dynamics), the loss is computed using a classifier-based loss function, typically cross-entropy, for each attribute:

$$\mathcal{L}_{\text{attr}} = \sum_{i=1}^4 \text{CrossEntropy}(A_i, \hat{A}_i)$$

where A_i and \hat{A}_i are the true and predicted values for the four musical attributes, respectively.

Step 3: Audio Encoder. As described in Section 3, the training data for the audio encoder is derived from MIDI files. Given a MIDI song S , we convert it to an audio waveform W . For each bar B_i in S and its corresponding waveform segment in W (denoted as B_w), we extract the note sequence from S and use the E-MLP to obtain an embedding sequence. Next, B_w is fed into the audio encoder, which outputs the note features and musical features. The bar decoder then uses the note features to reconstruct the embedding vector list, while the musical features are used to predict the four attributes (onset, duration, pitch, and velocity).

The loss function in this process consists of two parts: 1) MSE Loss: The Mean Squared Error (MSE) between the input embedding sequence from the E-MLP and the output embedding sequence generated by the bar decoder; 2) Classification Loss: A cross-entropy loss for the four musical attributes predicted from the musical features.

$$\mathcal{L}_{\text{audio}} = \frac{1}{T} \sum_{t=1}^T \text{MSE}(E_t, \hat{E}_t) + \sum_{i=1}^4 \text{CrossEntropy}(A_i, \hat{A}_i)$$

where E_t is the input embedding at time step t , \hat{E}_t is the reconstructed embedding, and T is the length of the sequence. In this training process, the bar decoder is frozen, and only the audio encoder is updated.

B DEALING WITH AUDIO INPUT

B.1 HARMONIC CONSTANT-Q TRANSFORMATION

The Harmonic Constant-Q Transform (H-CQT) extends the traditional Constant-Q Transform (CQT) by introducing a third dimension, designed to align harmonically related frequencies. This alignment allows the transformation to more effectively capture harmonic information, which is crucial in many audio analysis tasks, such as music transcription, pitch tracking, and timbre analysis. By analyzing harmonics across multiple frequency bins, the H-CQT emphasizes both the fundamental frequencies and their harmonics, improving the representation of harmonic structures.

The H-CQT is mathematically defined as follows:

$$X_{HCQT}(h, t, k) = \sum_{n=0}^{N-1} x(n) w(n-t) e^{-2\pi i h f_k n}$$

where:

- $X_{HCQT}(h, t, k)$ represents the H-CQT coefficient at harmonic index h , time frame t , and frequency bin k .
- $x(n)$ is the input signal in the time domain, typically a sampled audio signal of length N .

- $w(n-t)$ is a window function (such as a Hann or Hamming window) centered around time frame t , which serves to localize the analysis in both time and frequency domains. The window function is crucial in minimizing spectral leakage and ensuring that the signal is properly segmented in time.

- f_k denotes the center frequency of the k -th bin in the Constant-Q Transform, which is characterized by a logarithmic frequency spacing, making it well-suited for processing audio signals where perception of pitch is often logarithmic in nature.

- h is the harmonic index, which allows for the computation of higher harmonics of the fundamental frequencies present in the signal. Specifically, when $h = 1$, the transform focuses on the fundamental frequencies, while $h > 1$ captures the harmonics.

In this formulation, the summation operates over all signal samples $x(n)$, where $n = 0, 1, \dots, N-1$. The term $e^{-2\pi i h f_k n}$ is a complex exponential that shifts the signal by a frequency determined by both the harmonic index h and the center frequency f_k . This process is analogous to taking a short-time Fourier transform, but in the context of CQT, it scales the analysis resolution logarithmically across frequencies, ensuring that each frequency bin captures a musically relevant pitch range.

By introducing the harmonic index h , the H-CQT enables the simultaneous representation of multiple harmonics for each fundamental frequency. This multidimensional transform offers a richer description of harmonic content compared to the traditional CQT, making it a powerful tool for analyzing signals where harmonic structure is important, such as music.

B.2 TIME ALIGNMENT FOR THE SECOND OF A BAR

As discussed in the conclusion section, obtaining the BPM (beats per minute) information for a piece of audio is a crucial step. After acquiring this information, we extract a 2-second audio segment from the raw input and process it using the audio encoder and bar decoder to predict the time signature of the audio.

Let us assume the time signature is represented as $\frac{n}{m}$, where n is the number of beats per measure (or bar), and m defines the note value that constitutes one beat (e.g., quarter note, eighth note). Additionally, let the BPM of the audio be denoted as BPM.

The duration of one bar in seconds, which corresponds to the time it takes to complete a single measure, can be calculated using the following formula:

$$\text{Duration of one bar (in seconds)} = \frac{n}{\text{BPM}} \times 60$$

Where: - n is the number of beats per bar (from the time signature), - BPM is the beats per minute, - 60 is the number of seconds in a minute.

Using this calculated duration, we define the new segment length for processing the entire audio. The audio will now be divided into segments corresponding to the duration of one bar, allowing for more precise temporal alignment. This adjustment ensures that each segment represents a musically coherent unit, aligning with the beats and measures of the audio.

C INSTRUMENT SEPARATION

To simplify the classification of instruments, we propose a streamlined categorization based on their corresponding MIDI program IDs, as shown in Table 8. This approach consolidates multiple MIDI instrument categories into broader groups, facilitating more efficient handling in instrument separation tasks. For example, various types of pianos (e.g., acoustic grand piano, electric piano) are grouped under a single "Pianos" category. Similar consolidations are applied across other instrument families, such as chromatic percussion, guitars, and basses.

The primary objective of this reduction is to enhance the efficiency of instrument separation models by lowering the complexity of the classification process while preserving the key distinctions necessary for accurate audio representation and processing. Each generalized instrument group corresponds to a specific range of MIDI program IDs, with the exception of standard drums, which are categorized separately.

We categorize instruments into three main groups: Monophonic, Polyphonic, and Percussion. Below is the detailed classification for each category, as shown in Table 9.

Monophonic Instruments. Monophonic instruments are those that generally produce only one note at a time. These instruments are commonly used in melodic lines and solos, as their tonal clarity allows for strong emphasis on individual notes including: 1) Violin: A bowed string instrument known for its wide range and expressive dynamics. 2) Trumpet: A brass instrument with a bright and powerful sound. 3) Saxophone: A woodwind instrument with a reedy sound, common in jazz and classical genres. 4) Bassoon: A large woodwind instrument, characterized by its deep, rich sound. 5) Flute: A woodwind instrument producing sound by air flow across an opening, known for its bright and airy timbre.

Polyphonic Instruments. Polyphonic instruments can play multiple notes simultaneously, making them suitable for harmonic and chordal accompaniment. These instruments are essential in creating complex textures in music including: 1) Piano: A keyboard instrument capable of producing rich harmonies and complex melodies. 2) Guitar: A string instrument often used in various musical genres, capable of playing chords and melodic lines. 3) Organs: A keyboard instrument with

Table 8: Instrument reduction according to MIDI program IDs

#	Program ID	Instrument in UniComposer
1	0-7	Pianos
2	8-15	Chromatic Percussion
3	16-23	Organs
4	24-31	Guitar
5	32-39	Bass
6	40-43	Violin
7	54-58	Trumpet
8	62-65	Saxophone
9	66-69	Bassoon
10	70-77	Flute
11	102-109	Ethnic
12	110-117	Melodic Percussion
13	/	Standard Drums

multiple sound-producing pipes, known for its grandeur in churches and large venues. 4) Bass (Electric/Acoustic): A low-pitched instrument that plays a foundational role in harmonic structures, often contributing to both rhythm and harmony.

Percussion Instruments. Percussion instruments are classified based on their ability to produce sound through striking or hitting. This category includes both rhythmic and melodic percussion instruments including: 1) Melodic Percussion: Instruments such as xylophones and vibraphones that can produce pitched sounds, allowing for melodic content. 2) Chromatic Percussion: Instruments capable of producing all the notes in the chromatic scale, often used for melodic and harmonic roles in orchestras. 3) Standard Drums: Drums typically found in drum kits, primarily used for rhythmic purposes in various musical genres.

Table 9: Three category instrument separation.

Category	Instruments
Monophonic	Violin, Trumpet, Saxophone, Bassoon, Flute
Polyphonic	Piano, Guitar, Organs, Bass
Percussion	Melodic Percussion, Chromatic Percussion, Standard Drums

D DATASETS

D.1 NOTE VOCABULARY SELECTION

We collect 608,020 unique notes from LMD dataset. The rationale behind collecting all notes that occur in the LMD dataset is outlined as follows:

1) Practicality of Limiting the Scope: It is computationally impractical to account for every possible combination of notes, as the total number of potential notes within the five-attribute representation is exceedingly large. Specifically, this involves combinations of the following five attributes: instrument type, pitch, duration, velocity, and time signature. The total number of potential combinations is the product of the ranges of these attributes, calculated as 24 (onset) * 24 (duration) * 4 (velocity bin) * 128 (pitch) * 13 (reduced instrument), resulting in an immense number of possible notes. Given this, it is more feasible to focus on notes that are actually observed in the dataset, as they represent a more manageable subset of possibilities.

2) Infrequency of Certain Combinations: Many attribute combinations are highly unlikely or do not occur frequently in real-world compositions. For instance, specific instruments such as the violin rarely play pitches lower than G3 (pitch number 55). Therefore, it is unnecessary to consider these improbable combinations, further reducing the computational complexity by excluding unrealistic or rare note occurrences. This selective approach helps in focusing on the most relevant data.

D.2 BAR-LEVEL FEATURE EXTRACTION

The four attributes of a bar mentioned in this paper are chord, time signature, dynamics, and tonality. These attributes are extracted from a bar, given the five-attribute notes contained within it.

Chord. We adopt the chord calculation algorithm from the REMI representation. A chord is defined by three attributes: root, chroma, and bass. These can be extracted using the library `mir_eval` by analyzing the chord name. Given the sequence of note pitches in a bar, a similarity metric is computed between this pitch sequence and all possible chord chromas. The chroma with the highest similarity is selected as the most likely chroma for the current bar, and the corresponding chord is then determined. This method ensures the chord assigned to a bar accurately reflects the harmonic content of that segment.

Time Signature. In the MIDI standard, the time signature is embedded in the program change channel, where multiple time signatures can exist within a single piece of music. To determine the time signature of a specific bar, we record the index of each bar and its corresponding program change. The time signature of a bar is assigned based on the nearest preceding program change. This method ensures that each bar’s time signature is correctly aligned with the overall structure of the piece.

Dynamics. We categorize dynamics into six levels: `pp`, `p`, `mp`, `mf`, `f`, and `ff`, which correspond to very soft to very loud dynamic ranges. To determine the dynamics of a bar, we extract the velocity and pitch information from the note sequence in the bar. The average velocity of all notes in the bar is computed, and based on this value, we map the velocity to one of the six dynamic levels. For instance, very low velocities map to `pp`, while very high velocities map to `ff`. This approach provides a clear dynamic profile for each bar, reflecting the performance intensity.

Tonality. We adopt a simple binary classification of tonality: major and minor. Major tonality often expresses emotions such as happiness, brightness, and triumph, while minor tonality conveys sadness, tension, or somberness. In this paper, tonality is derived together with the chord extraction process. A chord label, such as ‘C:maj7’ or ‘D:min7’, directly indicates whether the tonality is major or minor. The label ‘maj’ (major) or ‘min’ (minor) is used to classify the tonality of the bar. This binary tonality annotation simplifies the tonal analysis of the music while still capturing essential emotional qualities.

D.3 HANDLING NOTE DURATION ACROSS BARS

In our approach, each bar is subdivided into 24 bins to represent discrete time intervals within the bar. However, a challenge arises when dealing with notes that extend beyond a single bar. Specifically, a note may begin in one bar but continue into subsequent bars, resulting in a duration that spans multiple bars.

To address this, we segment the duration of such notes into discrete parts. Each segment corresponds to a note that starts within a specific bar and, in most cases, extends to the end of that bar. The remaining portion of the note continues into the next bar in a similar manner. By this method, we ensure that each bar is represented by a set of notes with a maximum duration of 24 bins, effectively standardizing the duration representation within each bar.

This approach allows for seamless integration of note durations across multiple bars while maintaining the temporal structure imposed by the 24-bin division. It also facilitates the uniform handling of note events across bars, enabling consistent analysis and processing of musical sequences.

E METRICS DETAILS

We evaluate performance using the note-level F-measure (F), where a note is considered correctly identified if its pitch is within a quarter tone, the onset is within 50 ms, and the offset is within 20% of the note’s duration.

Chord Accuracy (CA). Chord Accuracy assesses whether the chords of the generated tracks align with the conditional chord sequence, which directly impacts the harmony of the generated music.

Chord accuracy is defined as:

$$CA = \frac{1}{N_{\text{tracks}} \times N_{\text{chords}}} \sum_{i=1}^{N_{\text{tracks}}} \sum_{j=1}^{N_{\text{chords}}} \mathbb{I}\{C_{i,j} = \hat{C}_{i,j}\}$$

where N_{tracks} represents the number of tracks, N_{chords} the number of bars, $C_{i,j}$ denotes the j -th ground-truth chord in the i -th track, and $\hat{C}_{i,j}$ refers to the corresponding generated chord in the same position.

Pitch, Velocity, Duration, Onset Interval. To provide a more comprehensive evaluation of the harmony, dynamics, and expressiveness of musical compositions, we analyze the distributions of various features (e.g., pitch and velocity) and compare the distances between the distributions of generated and ground-truth musical pieces. First, histograms are computed for each feature, followed by kernel density estimation to convert the histograms into continuous probability density functions. This smoothing process offers a more generalizable representation of the data.

Averaging Overlapped Area (OA). The overlapping area (OA) of distributions ($\mathcal{D}_{\mathcal{A}}$, where \mathcal{A} can be one of Pitch, Velocity, Duration, or Onset Interval) is used to quantify the difference between the generated and ground-truth musical pieces. This is represented by the following formula:

$$\mathcal{D}_{\mathcal{A}} = \frac{1}{N_{\text{tracks}} \times N_{\text{bars}}} \sum_{i=1}^{N_{\text{tracks}}} \sum_{j=1}^{N_{\text{bars}}} \text{OA}(\mathcal{P}_{i,j}^{\mathcal{A}}, \hat{\mathcal{P}}_{i,j}^{\mathcal{A}})$$

where OA denotes the averaged overlapping area between two distributions. $\mathcal{P}_{i,j}^{\mathcal{A}}$ represents the distribution of feature \mathcal{A} in the i -th track and j -th bar of the ground-truth musical piece, while $\hat{\mathcal{P}}_{i,j}^{\mathcal{A}}$ represents the same in the generated piece.

F MORE SHOWCASES AND DISCUSSIONS

F.1 FORMULATION OF THE CASES

We denote MIDI music as images, where the height of the image represents the pitch of each note, and the length corresponds to the time. In this representation, each note is mapped to a specific position in the image based on its pitch and onset time. The color of each pixel represents instrument, providing a visual representation of the musical structure. More showcases are on <https://sites.google.com/view/unicomposer>

F.2 BENEFITS OF USING BARS AS THE BASIC UNIT

Since UniComposer uses bars as the fundamental unit for music generation, the resulting compositions are strongly aligned by bars, which helps to maintain a coherent and well-structured musical form. This bar-level alignment ensures that the generated music follows a clear rhythmic structure, improving both the musicality and organization of the output.

F.3 BENEFITS OF INSTRUMENT SEPARATION

UniComposer distinguishes between monophonic, polyphonic, and percussion instruments, allowing for a more refined focus on their individual roles within band-level music compositions. Monophonic instruments primarily contribute to melodic variation and dynamic changes, playing a central role in driving the musical narrative. Polyphonic instruments provide harmonic support and stability, often through repetitive chord progressions or sustained notes that enhance the overall harmonic texture. Percussion instruments, on the other hand, serve as the rhythmic foundation, dictating the tempo and guiding the progression of the music. This separation of instrumental roles contributes to a clearer structural organization in band-level compositions, ensuring that each instrument type complements the others in a cohesive and balanced manner.

