Supplementary Material

A Data Preprocessing

In the preprocessing phase, songs in the MIDI collection are first filtered according to specific conditions, keeping only the ones that:

- Maintain a time signature of 4/4 throughout their entirety.
- Contain at least one non-empty drums track (MIDI channel 10).
- Contain at least one non-empty bass track (MIDI program number in the range [32, 39]).
- Contain at least one non-empty guitar/piano track (MIDI program number in the range [0, 31]).

After the filtering phase, each song is preprocessed in order to obtain sequences with a fixed number of bars and tracks. Since a song may contain several drum, bass and guitar/piano tracks, multiple subsongs are derived from it by computing a cross-product between the three classes of tracks. The rest of the tracks are merged into a single "strings" track which is appended to each combination resulting from the cross-product. At the end of this process, each subsong is composed of 4 tracks: a drum track, a bass track, a guitar/piano track and a strings track. Each MIDI subsong combination is then transformed into a pianoroll. In order to obtain fixed size sequences of music, denoting with N the number of bars, a sliding window of size N and stride 1 is slid along the bar axis of the subsong's pianoroll to extract the final samples. Data augmentation is performed on each subsequence by randomly transposing the pitch of all notes by a number of semitones uniformly sampled from the interval [-5, 6]. The resulting sequence of N bars is added to the final dataset only if it does not contain any bar of complete silence. Each sample is finally stored as a pair of tensors $(\mathbf{S}, \mathbf{X})$, where $\mathbf{S}$ and $\mathbf{X}$ are, respectively, the structure tensor and the content tensor associated to the musical sequence.

B Model

B.1 Convolutional Graph Network

In our model, the content encoder and the content decoder both use a Graph Convolutional Network (GCN) to propagate musical information. The GCN exploits the structure $\mathcal{S}$ of a chord-level graph g. The ℓ-th layer of the GCN aggregates the information contained in the neighborhood of each node v, computing new node states $\mathbf{h}_v^{\ell+1} \in \mathbb{R}^d$ as follows:

$$\mathbf{h}_v^{\ell+1} = \mathrm{ReLU}\left(\sum_{t\in\mathcal{T}} \sum_{u\in\mathcal{N}_v^t} \frac{1}{|\mathcal{N}_v^t|} \mathbf{W}_t^{\ell+1} e\left(\mathbf{h}_u^{\ell}, \boldsymbol{\delta}_{uv}\right) + \mathbf{W}^{\ell+1}\mathbf{h}_v^{\ell}\right) + \mathbf{h}_v^{\ell}, \tag{1}$$

where $\mathcal{T} = \{1, 2, \ldots, \theta\}$ is the set of edge types, $\mathcal{N}_v^t$ is the neighborhood of v restricted to edges (u, v) of type $\tau_{uv} = t$, $\boldsymbol{\delta}_{uv} \in \mathbb{R}^T$ is the one-hot distance in timesteps between u and v, $\mathbf{W}_t^{\ell+1} \in \mathbb{R}^{d\times d}$ and $\mathbf{W}^{\ell+1} \in \mathbb{R}^{d\times d}$ are learnable weight matrices and $e\colon \mathbb{R}^d \times \mathbb{R}^T \to \mathbb{R}^d$ is a learnable function defined as:

$$e\left(\mathbf{h}_u^{\ell}, \boldsymbol{\delta}_{uv}\right) = \mathrm{ReLU}\left(\mathbf{D}(\boldsymbol{\delta}_{uv}) \odot \mathbf{h}_u^{\ell}\right), \tag{2}$$

where $\mathbf{D} \in \mathbb{R}^{d\times T}$ is a learnable distance embedding matrix that transforms one-hot timestep distances $\boldsymbol{\delta}_{uv}$. The weight matrix $\mathbf{D}$ is shared across all the convolutional layers, forcing the network to find a single general representation for distances.

Looking at the first term in Equation 1, v's state $\mathbf{h}_v^{\ell}$ is first transformed through the weight matrix $\mathbf{W}^{\ell+1}$. Then, node states $\mathbf{h}_u^{\ell}$ coming from the neighborhood of v are transformed through the function e on the basis of the timestep distance $\boldsymbol{\delta}_{uv}$ between u and v. Finally, if $\tau_{uv} = t$, the modified node states are further transformed through the weight matrix $\mathbf{W}_t^{\ell+1}$ and scaled by $1/|\mathcal{N}_v^t|$. The resulting values are summed and passed through a ReLU activation function. The final node state $\mathbf{h}_v^{\ell+1}$ is obtained by summing this last value with the previous node state $\mathbf{h}_v^{\ell}$. This represents a residual connection between consecutive layers in the GCN.