APPENDIX A. EXPERIMENTS DETAILS

A.1 CNN-VAE

While the VQ-VAE maps data into the nearest embeddings of a 2-D dictionary, the CNN-VAE samples z from 1-D Gaussian distribution (latent_dim=512). It requires additional parameters for linear projections to 1-D Gaussian parameters and reconstruction to 2-D space. The design of encoder and decoder is same as the VQ-VAE. During training, AdamW optimizer is applied with cosine annealing from 1e-3 to 5e-6 for 500 epochs. Also, β annealing from 0 to 1 is applied.

A.2 Music Transformer

We adopt only decoder Music Transformer with relative global attention. The model hyperparameters are set to (num_layers=4, hidden_size=128, and num_heads=2). We have verified that larger models fail to be trained or improved. To process multi-label representation, input embedding layers (nn.Embedding) are replaced with linear layers (nn.Linear). During training, AdamW optimizer is applied with cosine annealing from 1e-3 to 5e-6 for 100 epochs. The training process works in teacher forcing with binary cross-entropy. At inference time, $p(x_0)$ sampled from the training set are used to start tokens in autoregressive mode. Each Sigmoid output of the model is taken as a parameter for Bernoulli distribution to realize stochasticity.

A.3 MuseGAN

We design two generators and two discriminators responsible for the two instruments. Except that, we follow experimental configurations of the original paper. During training, Adam optimizer is applied with a learning rate of 0.001. The whole model is trained with WGAN-GP loss. The generators are updated on every 5 updates of the discriminators.

APPENDIX B. IMPLEMENTATION DETAILS

B.1 Environments

Our implementation works on Python (==3.8.8), torch (==1.9.0), pytorch-lightning (==1.5.9), jupyterlab (==3.2.9), numpy (==1.20.1), scipy (==1.6.2), pretty-midi (==0.2.9), pyFluidSynth (==1.3.0), pypianoroll (==1.0.4), prdc (==0.2), and sparse (==1.6.2).

B.2 How to play pianoroll representation

As referred in [21], the time-grid based representation cannot distinguish between long notes and repeated notes. If notes are repeated successively, we simply regard them as a continuous one. For the human listening test, note velocity is set to 80 ($\pm\varepsilon$) for bass and 90 ($\pm\varepsilon$) for drum. Also, tempo is randomly chosen for each sample.

B.3 How to obtain correlation matrix

To obtain C_{WAV}, a 1-D audio x_{WAV} is divided into B bars and segments in each bar are transformed into mel-spectrograms using bar-length time steps. C_{MIDI} can be obtained as 1) computing the hamming distance between bars, 2) normalizing the distance matrix by its maximum value, and 3) subtracting the matrix from one to express correlation.

B.4 Network Architectures

We introduce network architectures of the VQ-VAE and their parameters used for the experiments. The model is based on basic convolution blocks consisting of (conv1d-batchnorm-leakyrelu(0.2)). In tables, an operation of convolution blocks is denoted in the form of conv(channel, kernel_size, padding, stride). Similarly, residual blocks are denoted as res(channel, kernel_size, padding, stride).

Input: $x \in \{0,1\}^{128\times 57}$

Model	Operations	Output Shape
Encoder	x = conv(128, 3, 1, 1)(x)	x shape: (128, 128)
	x = conv(128, 3, 1, 2)(x) x = res(128, 3, 1, 1)(x)	x shape: (64, 128)
	x = conv(128, 3, 1, 2)(x) x = res(128, 3, 1, 1)(x)	x shape: (32, 128)
	x = conv(16, 3, 1, 1)(x)	x shape: (32, 16)
Quantize	x = quantize(x)	x shape: (32, 16)
Decoder	x = conv(128, 3, 1, 1)(z)	x shape: (32, 128)
	x = upsample(2)(x) x = conv(128, 3, 1, 1)(x) x = res(128, 3, 1, 1)(x)	x shape: (64, 128)
	x = upsample(2)(x) x = conv(128, 3, 1, 1)(x) x = res(128, 3, 1, 1)(x)	x shape: (128, 128)
	x = conv(57, 3, 1, 1)(x)	x shape: (128, 57)
	x = Sigmoid(x)	x shape: (128, 57)

Output: $x \in \{0,1\}^{128\times 57}$

For an autoregressive prior, we adopt LSTM with 4 layers of which the size of hidden states is 512. First, the sequences of indices from the VQ-VAE are embedded to 128 dimensional vectors (nn.Embedding) and they are fed to the LSTM. Each of the LSTM outputs is got through a linear layer (nn.Linear) to match the vocab size.

APPENDIX C. LOOP DETECTOR ANALYSIS

C.1 Analysis of the loop detector

Figure 7 indicates the loop score for the training and validation set of C_{WAV}. Some outliers can be observed for the validation set.

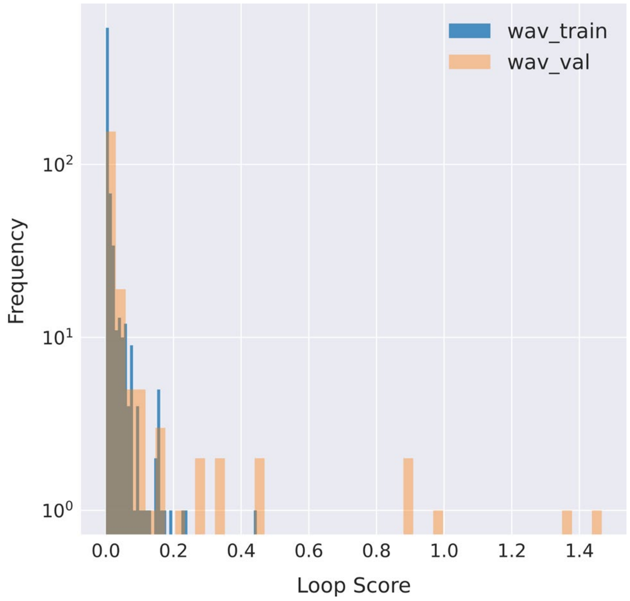

Figure 7. The loop score for training and validation set of C_{WAV}. The frequency at the y-axis is represented in log scale.

Figure 8 indicates the loop score of all C_{MIDI} in Lakh MIDI Dataset. Compared to C_{WAV}, many samples are out of our threshold (≈0.01).

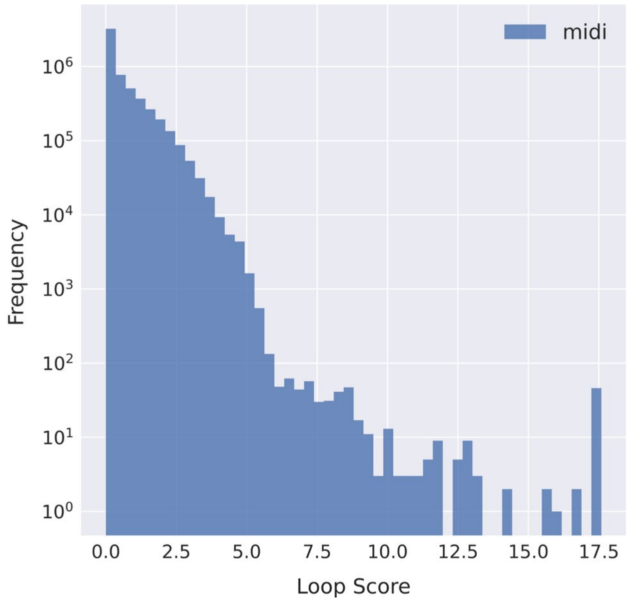

Figure 8. The loop score of C_{MIDI}. The frequency at the y-axis is represented in log scale.