Appendix
Quantized GAN for Complex Music Generation from Dance Videos

Ye Zhu[1⋆], Kyle Olszewski[2], Yu Wu[3], Panos Achlioptas[2], Menglei Chai[2], Yan Yan[1], and Sergey Tulyakov[2]

[1] Illinois Institute of Technology, USA
[2] Snap Inc., USA
[3] Princeton University, USA

1 Network Architecture

1.1 Generator

The following Table 1, Table 2, Table 3 and Table 4 show the detailed model architectures of the motion encoder, high-level VQ generator, low-level VQ generator and the residual block, respectively.

Table 1. Architecture for the motion encoder.

6 × 1, stride=1, Conv 256, LeakyReLU
Residual Stack 256
3 × 1, stride=1, Conv 512, LeakyReLU
Residual Stack 512
3 × 1, stride=1, Conv 1024, LeakyReLU
Residual Stack 1024
3 × 1 , stride=1, Conv 1024, LeakyReLU
4 × 1, stride=1, Conv 1

1.2 Discriminator

We adopt the multi-scale discriminator design for the proposed *D2M-GAN*, where is formed by a stack of 3 discriminator blocks that operates on the original VQ sequence, and its downsampled features based on the window-based objective functions as introduced in the main paper. The architecture of each discriminator block is shown below in Table 5.

⋆ This work was mainly done while the author was an intern at Snap Inc.

Table 2. Architecture for the high-level VQ generator.

6 × 1, stride=2, Conv 32, LeaklyReLU
Residual Stack 32
41 × 1, stride=2, Conv 64, LeakyReLU
Residual Stack 64
41 × 1, stride=1, Conv 128, LeakyReLU
Residual Stack 128
41 × 1, stride=1, Conv 256, LeaklyReLU
Residual Stack 256
41 × 1, stride=1, Conv 512, LeakyReLU
Residual Stack 512
40 × 1, stride=1, Conv 64
Tanh()

Table 3. Architecture for the low-level VQ generator.

6 × 1, stride=2, Conv 32, LeakyReLU
Residual Stack 32
4 × 1, stride=1, Conv 64, LeakyReLU
Residual Stack 64
40 × 1, stride=2, Conv 128, LeakyReLU
Residual Stack 128
40 × 1, stride=1, Conv 256, LeakyReLU
Residual Stack 256
40 × 1, stride=1, Conv 512, LeakyReLU
Residual Stack 512
40 × 1, stride=1, Conv 1024, LeakyReLU
Residual Stack 1024
40 × 1, stride=1, Conv 1024, LeakyReLU
40 × 1, stride=1, Conv 64, LeakyReLU
Tanh()

2 Experimental Details

We implement the entire framework using the PyTorch [5] framework for automatic differentiation and GPU-accelerated training and inference.

Pre-learned Codebook. We adopt two independently pre-trained codebooks for two levels in our *D2M-GAN*. Specifically, the original JukeBox [1] contains three levels of VQ-VAE [4] based models, which are defined as top, middle and bottom levels with hop lengths of 128, 32, and 8, respectively. We adopt the top level codebook for the high-level *D2M-GAN*, and the middle level codebook for the low-level *D2M-GAN*. Therefore, for a two-second audio sequence with a sampling rate of 22050 Hz, the generated VQ sequences from the high-level and low-level VQ generators are in dimension of 64×344 and 64×1378, respectively, where 64 is the dimension of the codebook entry, 344 and 1378 are the sequence

Table 4. Architecture for the residual stack.

LeakyReLU, dilation=1, Conv
LeakyReLU, dilation=1. Conv
Shortcut Path
LeakyReLU, dilation=3, Conv
LeakyReLU, dilation=1, Conv
Shortcut Path
LeakyReLU, dilation=9, Conv
LeakyReLU, dilation=1, Conv
Shortcut Path

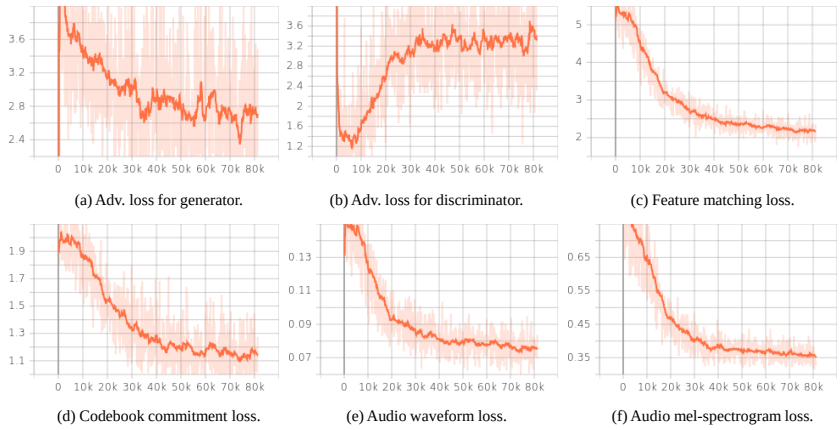

Fig. 1. Training losses for the proposed *D2M-GAN*. *Adv.* stands for adversarial.

lengths. It is worth noting that the hop length is a key factor that influences the trade-off between the generated audio quality and model scale in general audio generation works. Specifically, a larger hop length represents higher compression and abstraction ability in the bottleneck layers with codebooks, but leads to relatively high level of noises in the synthesized musical samples. Actually, only the bottom level from the original JukeBox model [1] is able to generate music with high audio quality, however, it takes *3 hrs* to sample a *20s* musical sequence, which is extremely time-consuming. Considering the primary goal of our task, which is to the capture the correlations between dance input and music output, we only test the model with hop lengths of 128 and 32 in our main experiments.

Training Losses. Since our proposed *D2M-GAN* includes multiple loss terms in the overall training objective, we show the change of each loss term during the training process in Figure 1. It is worth noting the model architectures and techniques described in our main paper are crucial for *D2M-GAN* to maintain a stable training. Notably, the codebook commitement loss, audio waveform loss

Table 5. Architecture for the discriminator block.

15 × 1, stride=1, Conv 16, LeakyReLU
41 × 1, stride=4, Groups=4, Conv 64, LeakyReLU
41 × 1, stride=4, Groups=16, Conv 256, LeakyReLU
41 × 1, stride=4, Groups=64, Conv 1024, LeakyReLU
41 × 1, stride=4, Groups=256, Conv 1024, LeakyReLU
5 × 1, stride=1, Conv 1024, LeakyReLU
3 × 1, stride=1, Conv 1

and audio mel-spectrogram loss can reach the comparable levels with the GT audio samples after convergence.

3 TikTok Dance-Music Dataset

The current version of our TikTok dance-music dataset contains in total 445 videos, which we annotate from 15 TikTok dance video compilations. There are 85 different songs, with majority of videos having a single dance performer, and a maximum of five performers. The average length of each video is approximately 12.5s. We split the training and testing set based on the music IDs, and ensure that there are no overlapping songs for two splits.

Compared to the existing music and dance datasets such as AIST++ [6, 3], our dataset is closer to the real-world scenario with various background, which is also our initial motivation to introduce this dataset. Additionally, majority of the current datasets available are not initially proposed for the dance to music generation task, AIST [6] is designed for dance music processing, AIST++ [3] provides the extra annoations for the subset of AIST for generating dance motions conditioned on music, some other similar datasets for motion generation have also been introduced [2]. Therefore, we hope that our proposed TikTok dance-music dataset can serve as a starting point for relevant future researches.

4 Subjective Evaluations

We conduct the Mean Opinion Scores (MOS) test for the subjective evaluations. In total, 26 subjects participated our MOS tests, among which 9 of them are females, the rest are males.

Two of our music evaluation protocols are based on the human subjective evaluations, which are the dance-music coherence test and the music overall quality test. For the dance-music coherence test, each evaluator is asked to rate 15 dance videos that are post-processed by fusing the original visual frames and generated music samples from different models. Specifically, the evaluators are asked to rate from the coherence aspect of the dance video (*i.e.*, whether they feels the music is coherent with the dance moves) with reference to the GT videos and original music. For the overall quality test, 15 audio samples (without video

frames) are played during the test for each evaluator, after which the evaluator is asked to rate the sound quality from the score range of 1 to 5. It is worth noting that for the overall quality test, we do not compare with the music samples obtained from the symbolic MIDI representation based methods. This is due to the reason that the symbolic representations and pre-defined music synthesizers in nature do not introduce audio noises to the generated signals, which makes the music samples sound rather "clean and high-quality", while the continuous or VQ audio representations can hardly achieve the similar effects with a learned music synthesizer (samples included in our demo video). Therefore, we do not include the MIDI-based methods as our baselines for fairness considerations.

References

1. Dhariwal, P., Jun, H., Payne, C., Kim, J.W., Radford, A., Sutskever, I.: Jukebox: A generative model for music. arXiv preprint arXiv:2005.00341 (2020)
2. Lee, H.Y., Yang, X., Liu, M.Y., Wang, T.C., Lu, Y.D., Yang, M.H., Kautz, J.: Dancing to music. In: NeurIPS (2019)
3. Li, R., Yang, S., Ross, D.A., Kanazawa, A.: Ai choreographer: Music conditioned 3d dance generation with aist++. In: ICCV (2021)
4. Oord, A.v.d., Vinyals, O., Kavukcuoglu, K.: Neural discrete representation learning. In: NeurIPS (2017)
5. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., dÁlché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
6. Tsuchida, S., Fukayama, S., Hamasaki, M., Goto, M.: Aist dance video database: Multi-genre, multi-dancer, and multi-camera database for dance information processing. In: Proceedings of the 20th International Society for Music Information Retrieval Conference, (ISMIR) (2019)