A GAN Model With Self-attention Mechanism To Generate Multi-instruments Symbolic Music

Faqian Guan
College of Mathematics and Computer Science
Fuzhou University
Fuzhou, China
faqianguan@gmail.com

Chunyan Yu*
College of Mathematics and Computer Science
Fuzhou University
Fuzhou, China
therica@fzu.edu.cn

SuqiongYang
College of Mathematics and Computer Science
Fuzhou University
Fuzhou, China
Eve_yang_07@outlook.com

***Abstract*—GAN has recently been proved to be able to generate symbolic music in the form of piano-rolls. However, those existing GAN-based multi-track music generation methods are always unstable. Moreover, due to defects in the temporal features extraction, the generated multi-track music does not sound natural enough. Therefore, we propose a new GAN model with self-attention mechanism, DMB-GAN, which can extract more temporal features of music to generate multi-instruments music stably. First of all, to generate more consistent and natural single-track music, we introduce self-attention mechanism to enable GAN-based music generation model to extract not only spatial features but also temporal features. Secondly, to generate multi-instruments music with harmonic structure among all tracks, we construct a dual generative adversarial architecture with multi-branches, each branch for one track. Finally, to improve generated quality of multi-instruments symbolic music, we introduce switchable normalization to stabilize network training. The experimental results show that DMB-GAN can stably generate coherent, natural multi-instruments music with good quality.**

Keywords—symbolic music generation, Generative Adversarial Networks, multi-instruments, switchable normalization, self-attention mechanism

I. Introduction

Music generation has been considered as one of most exciting tasks in the field of music information retrieval. From the performing instruments perspective, music can be divided into two categories, single-instrument music and multi-instruments music. The multi-instruments music, usually played by different instruments, is more common. For example, to composite rock music, several instruments, containing guitars, electric guitars, basses, and drums, are necessary and always are the dominant instruments. Ooccasionally, piano and violin are also used as auxiliary instruments.

Compared to single-instrument music, generation for multi-instruments music is more challenging. First, music is not only an art of sound, but also an art of time. It means that temporal structure is an important issue for coherence of music. Hence, a mechanism to address temporal structure is critical but challenging. Second, different instruments are dependent, that is, they interact with each other. It means harmony is another important issue for melodiousness of multi-instruments music. From the computer processing perspective, multiple instruments can be perceived as multiply tracks, each track for one instrument. As a result, to simplify multi-instruments music generation, some prior works merely combined several independent tracks, without regard to interaction among different tracks, and resulted in poor harmony. Therefore, an architecture to coordinate various instruments over time is fundamental.

AI had made great progress in music generation. Walter et al. presented a music generation model based on HMM [1], with learned parameters of Markov chain and hidden Markov model by training data. Hadjeres et al. proposed a VAE-based music generation method and proved its effectiveness in monophonic music generation [2]. Especially, considering that music is closely related to time, RNN, a neural network for sequence data processing, has been widely employed in music generation. Eck et al. used two LSTM models to create music [3]. Sturm et al. proposed a recursive network to composite music in digital format [4]. Jaques et al. used RNN and reinforcement learning to train music generation models [5]. However, all generated music of above methods is far from satisfying to human ears.

In recent years, GAN [6] has been widely used in various generation tasks, such as image generation with specific style, and has always achieved state-of-the-art performance. Inspired by the exciting progress in image and video related tasks, some researchers had employed GAN to extract temporal features and applied them to music generation. Yang et al. proposed a convolution GAN [7] with specified conditions in 1D and 2D during generation process, can effectively solve the problem of lack of context association in music generation. Considering the temporal structure of music, [8,9] combined RNN and GAN to generate music. References [9,10] combined reinforcement learning with GAN for music generation. References [11,12] proposed a multi-track music notes generation model. Although these GAN-based methods have also achieved state-of-the-art performance in the field of multi-instruments music generation, harmony and coherence of generated music are not good enough, partly because the above methods only used convolution to extract features, which cannot extract the temporal features effectively.

In this paper, we propose a GAN model with self-attention mechanism [13] to generate multi-instruments symbolic music. The self-attention mechanism is integrated to help GAN- based network to extract both spatial and temporal features effectively.

This work was supported by the Fujian Foundation of Science 2018J01794 and the Fujian Health-Education Joint Project WKJ2016-2-26.

978-1-7281-2009-6/$31.00 ©2019 IEEE

Furthermore, DMB-GAN, a Dual Multi-Branches GAN architecture, one branch for an instrument, is employed to coordinate various instruments over time. Moreover, we employ SN [14] instead of BN [15] to reduce the impacts of gradient explosion and gradient disappearance partly due to self-attention mechanism. The experimental results show that DMB-GAN can be trained stably, leading to generate more consistent multi-instruments symbolic music with high quality.

II. Fundamentals

A. Hierarchical Structure of Music

To automatically generate music through GAN model, we should have a certain understanding of the hierarchical structure of music. In this hierarchical structure, higher-level building blocks are always made up of smaller recurrent patterns [11]. Generally, a song consist of multiple paragraphs, a paragraph is made up of multiple phrases, a phrase contains multiple bars, and a bar has multiple beats. From the music theory perceptive, beat is the basic unit for a song while bar is the common compositional unit. Fig. 1 depicts the hierarchical structure of music.

In symbolic music generation, given specific temporal resolution, a beat is divided into corresponding time steps, also called pixels, to cover common temporal patterns, such as 16th notes. For example, a beat in 4/4 times music is divided into 24 pixels in [11, 12] and 4 pixels in [7, 16, 17].

B. Piano-roll Represented Symbolic Music

Piano-roll representation is widely used in multi-instruments music generation [11]. Given an instrument, its corresponding piano-roll is defined as a binary-valued matrix, liking a score sheet, to represent the presence of note over time steps. The vertical axis represents note pitch and the horizontal axis for time. For multiple instruments, multiple piano-rolls can be defined, each for one instrument.

C. The Basic Idea of GAN-based Music Generation

Generally, the basic idea behind a GAN-based music generation model is similar to those of other generation models for image tasks. As shown in Fig. 2, randomized noise (z) is used as input data. The generator generates data similar to samples in target domain (x) so that the discriminator can't distinguish whether it is real data or generated data. The overall model is optimized through the minimax game. The data here may be a note, a mel energy, or a spectrogram extracted from music.

III. Method

A. Basic Generator Architecture for Single-instrument

Fig. 3 depicts the basic generator architecture for single-instrument music. In Fig. 3, shared data denotes input data. For a given input, we use two sequential extension dimensions methods. One is expanding through the time axis of priority, and the other is expanding through the pitch axis of priority. Finally, the output of the two methods is concatenated into a tensor. Because the channel of the single track is 1, we perform another deconvolution operation on the tensor to reduce the channel dimension to 1.

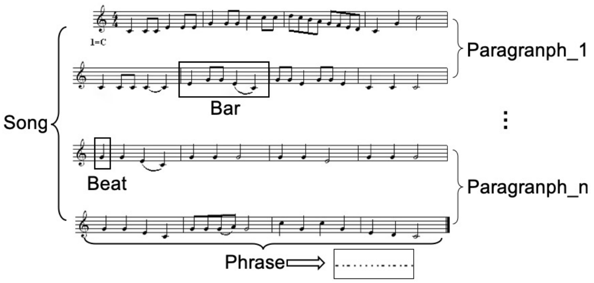


Fig. 1. Hierarchical structure of music.

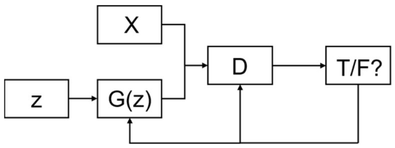


Fig. 2. GAN-based music generation.

The input of the single track is a common vector of 16*7*128, where 16 corresponds to the pixel number in a bar, 7 corresponds to the music parts, and 128 is the number of convolution kernels. Then, the music time axis and pitch axis are expanded to generate a single track music with a shape of 64*84*1, respectively, where 64 refers to the length of a phrase on the time axis and 84 refers to a pitch commonly used for music.

Our training data representation is in binary format while the output after deconvolution is in real-valued format. Hence, a residual block connecting binary neurons is appended to convert generated real-valued outputs to binary outputs, which denote single track music notes in piano-roll representation.

B. Imporved Generator Architecture with Self-attention Mechanism for Single-instrument

The basic generator architecture adopts convolution to obtain high-level features. Convolution can extract spatial features well. But it is difficult for convolution to extract temporal features effectively.

For data represented in piano-roll format, convolution merely extracts features of a small range of consecutive time steps and adjacent pitches. However, musical notes usually have long-term dependent characteristics. That is, some notes of a certain period of time steps usually have a certain relationship with those notes of a longer range. In addition, it is normal that the pitches of adjacent time steps may changes greatly. For example, the pitch of current time is E3 and the pitch of the following time may be D5. Unfortunately, convolution kernels cannot contain such two pitches due to limitation of convolution size. Thus, ordinary convolution fails to extract the temporal features of music. SeqGAN [9] tried to extract temporal features by RNN. However, SeqGAN is not suitable for multi-instruments symbolic music generation in piano-rolls format. Because piano-rolls for multi-instruments symbolic music is represented as high-dimensional data, and SeqGAN is difficult to train with high-dimensional data.

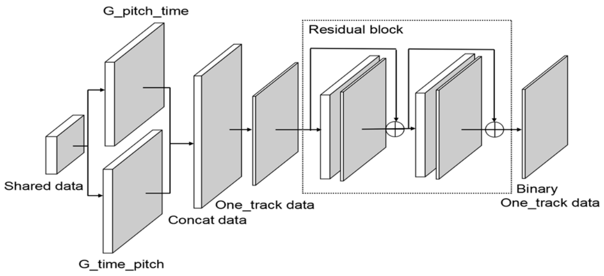


Fig. 3. Basic generator architecture for single-track.

To extract the temporal features of music, we introduce a self-attention mechanism [13] in our model. A self-attention module is deployed before the original generator to pay attention to different temporal characteristics. Fig. 4 describe the details about our self-attention module. First, we extract the original data into three feature spaces f(x), g(x), and h(x) by 1*1 convolution. Next, after transposed, f(x) is multiplied by g(x). The results are normalize with softmax to get an attention map. Then, the attention map is multiplied by h(x), pixel by pixel, to build a feature map of the final adaptive attention. Finally, corresponding to the basic model, the feature map is generated through G_pitch_time, G_time_pitch, and subsequent operations. In Fig. 4, the horizontal axis of example represents time, the vertical axis represents pitch, and the colored portion denotes attention area.

With the self-attention mechanism, it is possible to extract the time series features of a wide range or even a discontinuous time pitch. For notes like E3 and D5, which differ greatly in pitch over adjacent time steps, the model with self-attention mechanism can extract its temporal features effectively.

C. Dual Multi-Branches Architecture with Switchable Normalization for Multi-instruments

Our final GAN based multi-instruments music generation model is shown in Fig. 5, which has dual GAN architecture and multi-branches.

For each instrument, there is a branch, including a basic generator with self-attention module. From the perspective of music theory, a conductor is necessary to direct different instruments playing simultaneously to produce harmonious music. Hence, G is a trunk generator for all instruments, through which Z generates a common vector, which is the shared data in our basic architecture. Because the generators of different instruments share the share data, it can be considered that there is a certain relationship among the piano-rolls generated by each generator in every branch.

In Fig. 5, G1-G5 represent the respective generators of the five instruments, and shared data are inputted into the generator of each track to generate the respective track music. X1-X5 are the real-valued output of G1-G5. Through the residual block with binary neurons, the DMB-GAN finally generates binary data. The binary neurons are derived from the residual unit of Res-net [20], which is represented by R1-R5. X1-X5 is the final output of each instruments. They are concatenated to generate the final multi-instruments music.

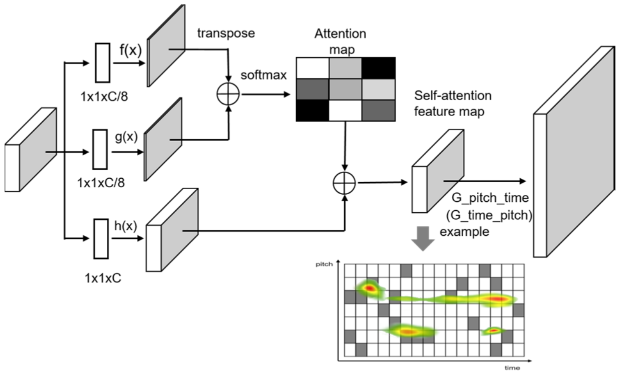


Fig. 4. Generator architecture with self-attention mechanism for one-track.

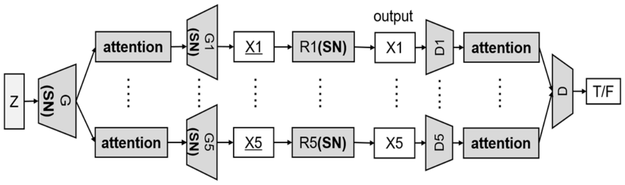


Fig. 5. Overall model with switchable normalization for multi-track.

Especially, in Fig. 5, SN denotes switchable normalization. It selects appropriate normalization through assigning different weights in BN, IN [18], and LN [19], making the model more stable to improve generated quality of music.

IV. Data preprocessing and Implementation Details

A. Data Preprocessing

The Lakh MIDI dataset (LMD) [21] is a collection of 176,581 unique MIDI files. Since the dataset contains a lot of noisy, we first match it with Million Song Dataset [22] for noise reduction and get 115,160 valid Midi files. Furthermore, we select 1384 midi files, which are all 4/4 beat music with rock tags, and with the highest matching confidence score.

As mentioned above, a beat of 4/4 time music can be subdivided into multiple pixels. According to [7,16,17], we divide a beat into four pixels, i.e. a pixel denotes a 16th note. Actually, 16th note is the most frequently used note in the music composition.

Piano-rolls format is a kind of binary representation with a vertical axis representing pitch. Since the pitches higher than C8 or lower than C1 are uncommon, in this paper, we only keep the pitches between C1 and C8, with a value of 84. Moreover, some auxiliary instruments only have few notes in the whole song. To reduce data sparsity, we merge these sparse tracks into a single piano-roll. Finally, we obtain our data all with 5 tracks, including Drums, Piano, guitar, Bass and other. Like [11,12,16], our goal is to generate a phrase of music, that is 4 consecutive bars. Thus, the resulting samples are of size 64*84*5.

The whole process of data processing is shown in Fig. 6.

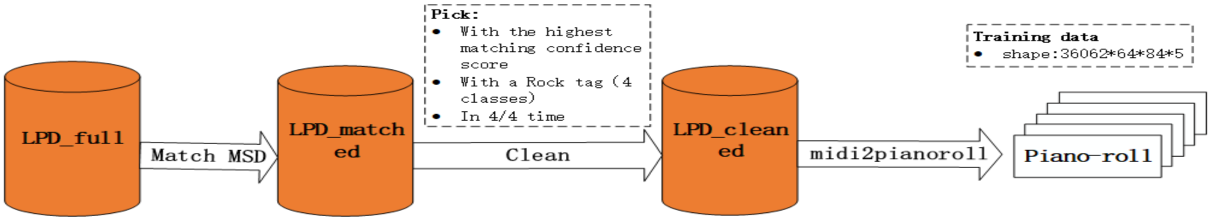


Fig. 6. Data processing flow chart.

B. *Architecture Parameters*

We use Adam optimizer [23] with an initial learning α=0.002. The training is divided into two phases. The first training phase do not include the residual block, and the second phase includes the residual block to generate the binary data directly. Both phases are trained for 20 rounds and a total of 40 rounds. The shape of the output data of the model is 64*84*5. In terms of batch size, we used 16 and 32 respectively.

V. Experiment and Results

A. *Objective Evaluation Metrics*

Following [11] and [12], we use empty bar rate, polyphonicity, note in scale, and a series of qualified note rate as our objective evaluation metrics.

Empty bar rate(EB) calculates ratio of empty bars. It indicates the continuity of the generated music.

Polyphonicity(PP) calculates the ratio of the number of time steps, where more than two pitches are played, to the total number of time steps. It indicates the polyphony of the generated music.

Note in scale(NS) calculates the ratio of the white key to the sum of the black and white keys in the piano-rolls format data. It can be used to roughly judge whether the generated music has the same style as that of training data.

Qualified note rate(QN) calculates the ratio of qualified note. We consider qualified note as 16th, 8th, 4th, 2nd, and 1st. Accordingly, we subdivide qualified note rate into qualified note rate 16(QN16), qualified note rate 8(QN8), qualified note rate 4(QN4), qualified note rate 2(QN2), qualified note rate 1(QN1), which represent the proportion of 16th note, 8th note, 4th note, 2th note, and whole note. If QN is low, the generated music is fragmented. Hence, this metric can indicate whether the music sounds coherent and natural.

For each trail model, we generate 1600 samples and calculate mean of each metric. A trail model with metrics that are the closet to those of our training datasets achieves excellent performance.

B. *Experimental Setups and Results*

MuseGAN [11,12] achieves state-of-the-art performance in multi-instruments symbolic music generation. Hence, we select MuseGAN with batch size 16 as our comparison model.

We use our DMB-GAN model without SN and self-attention module as the baseline, which is denoted as “Base”. The baseline use BN instead of SN. Also, our DMB-GAN model with SN and without self-attention module, denoted as “Switch”, is also selected as comparison model. A complete DMB-GAN model with SN and self-attention module, denoted as “Att_sw”, is the last comparison model.

Moreover, to illustrate the impact of batch size, we also execute comparison experiments with different batch sizes, 16 and 32.

TABLE I and Fig. 7 show the comparison results. Because the models with SN have little change when the batch size changes, only the result the complete DMB-GAN model with SN and self-attention module with batch size 32 is included in TABLE I. Fig. 8 depicts an instance of a generate music phrase. There are some samples generated by DMB-GAN model on our website[1].

C. *Result Analysis*

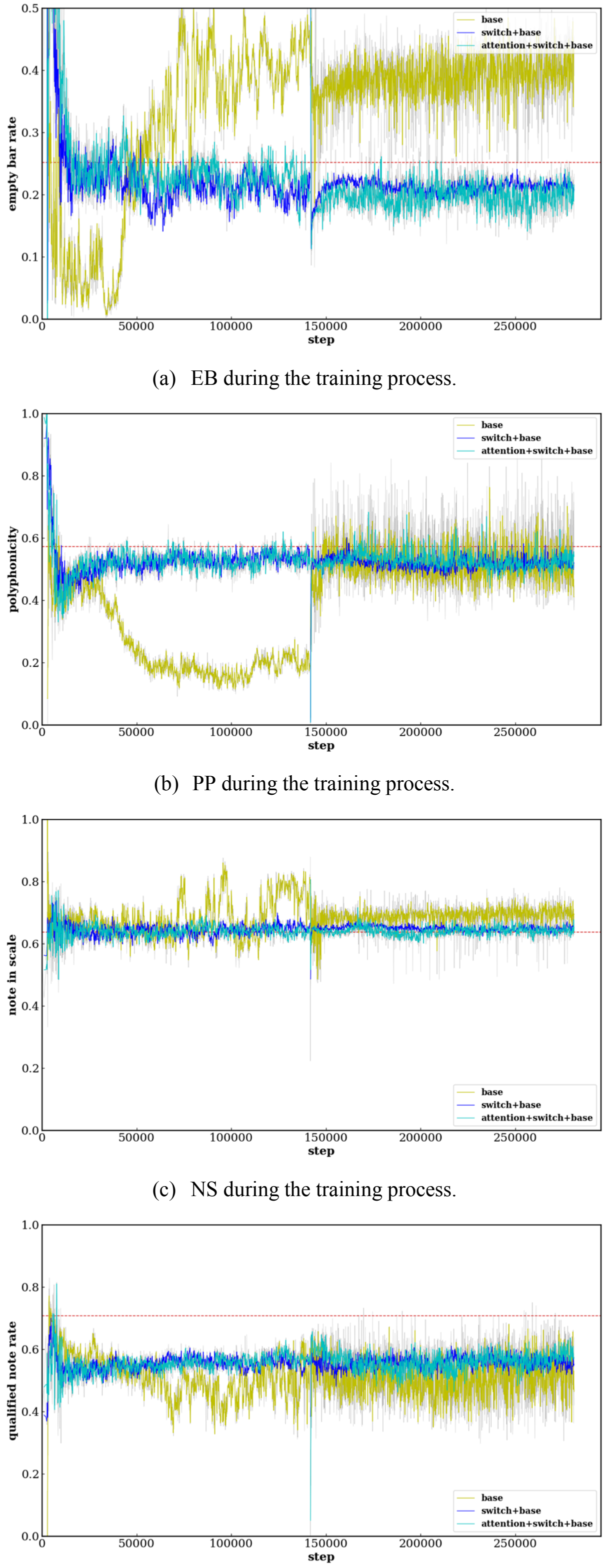
a) Self-attention Module Helps Extracting Temporal Features to Improve Music Generation Quality.

As shown in TABLE I, on all metrics, our DMB-GAN model with the self-attention module is closer to the training data than MuseGAN and all other comparison models. Compared to the training data, on metrics of NS, QN2, QN1, there are almost no difference, which is only different in the thousandth place.

TABLE I. Comparison Results. Musegan_batch16 is the result of Musegan with a batch size of 16. Base_batch16 is the result of DMB-GAN without with a batch size of 16. Base_batch32 is the result of our basic experiment with a batch size of 32. Switch_batch16 is the result of our introduction of SN in the base model with a batch size of 16. Switch_batch32 is the result of our introduction of SN in the base model with a batch size of 32. Att_sw_batch32 is the result of our final model that introduces SN and self-attention with a batch size of 32.

	EB	PP	NS	QN16	QN8	QN4	QN2	QN1
Training data	0.252	0.572	0.637	1	0.706	0.394	0.216	0.093
Musegan_batch16	0.211	0.504	0.650	0.446	0.205	0.087	0.028	0.006
Base_batch16	0.356	0.387	0.652	1	0.466	0.189	0.059	0.010
Base_batch32	0.353	0.498	0.701	1	0.543	0.287	0.107	0.018
Switch_batch16	0.227	0.486	0.637	1	0.580	0.296	0.117	0.044
Switch_batch32	0.220	0.506	0.653	1	0.557	0.329	0.170	0.055
Att_sw_batch32	**0.221**	**0.558**	**0.631**	**1**	**0.619**	**0.360**	**0.217**	**0.088**

[1] https://guanfaqian.github.io/dmbgan/



(a) EB during the training process.

(b) PP during the training process.

(c) NS during the training process.

(d) QN8 during the training process.

Fig. 7. Objective evaluation metrics during training process. The yellow line represents the base DMB-GAN model, the light blue line for DMB-GAN model with SN method, the dark blue line for the complete DMB-GAN model, and the red dotted line indicates training data.

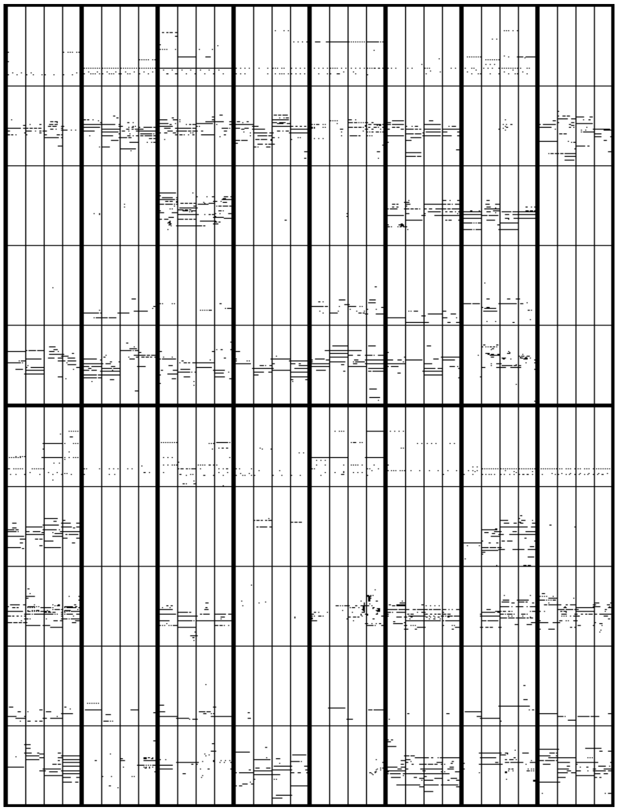


Fig. 8. Instance of a generate music phrase

on the remaining metrics of EB, PP, QN8, and QN4, the gap compared to training data is also less than that of the SN method alone.

Fig. 7 shows the entire training process. It can also be seen that after introducing the self-attention mechanism, the quality of the generated music has some improvement over the basic model and the model only with SN. Therefore, the self-attention mechanism is effective in DMB-GAN, which enables the model to generate more consistent and natural music with high-quality.

b) Switchable Normalization Method Helps to Stabilize Model Training to Improve Music Generation Quality.

As shown in Fig. 7, compared with the basic model using the BN method, our DMB-GAN model with SN of batch size 32 has great progress in stability. After the GAN convergence, the values of the SNs can be almost stabilized on one line, and the amplitude after convergence is much smaller than that of the BN method. More importantly, in TABLE I, the final result of each metric, obtained by SN is closer to the value of training data than BN, which indicates that the music generated by SN method is more consistent with the training data distribution.

Additionally, according to the characteristics of BN, our DMB-GAN model with BN is very sensitive to the settings of batch size. The experimental results shows that the effect will rise and then stabilize along with the increase of the batch size. However, due to the complexity of the model and data, many experiments cannot obtain an appropriate batch size in advance. After replacing BN with SN, we also performed a comparison experiment with batch size 32 and 16. As shown in TABLE I, we find that the results are not with much distinction. From this perspective, SN, which can achieve stable experimental results under different batch sizes, is superior to BN.

In short, after the introduction of SN, our model stability has been greatly improved, the quality of generated music is better than the original, and the same experimental results can be achieved under different batch sizes.

c) Low Temporal Pesolution Per Beat Helps Deduce Frangementation of Generated Music.

As shown in TABLE I, the multi-instruments music generated by our base model with BN is higher than the training data value on the EB, and the MuseGan is lower than the training data. However, after having removed the influence of GAN instability, the significant difference between them are temporal resolution per beat. MuseGAN defines a high temporal resolution that ours.

A higher EB value indicates that our method generates fewer notes, resulting in a polyphony metric of about 0.11 lower than that of the MuseGAN. Obviously, from the perspective of qualified note rate, in QN16, QN8, QN4, QN2, QN1, the music generated by our model is about twice as high as the effective note of MuseGAN. This means that the musical notes we generate are less changed in a short period of time, and can avoid been over-fragmented, making the music sound more coherent and natural.

Furthermore, training each batch of MuseGAN consumes about 2.66s while that of our model only about 0.58s, due to complex input data and model of MuseGAN caused by its higher temporal resolution per beat, which is about 5 times of ours.

In summary, our DMB-GAN model is more efficient and more natural than MuseGAN.

VI. Conclusion

In this paper, we propose a new GAN-based multi-instruments symbolic music generation architecture. The architecture involves dual generators with multi-branches, one branch to an instrument. The trunk generator acts as a conductor to coordinate various instruments over time. And the self-attention mechanism is deployed before every branch generator for each instrument to help GAN-based network to extract both spatial and temporal features effectively. Moreover, SN is introduced to reduce the impacts of gradient explosion and gradient disappearance. Thus, DMB-GAN, a Dual Multi-Branches GAN architecture, can be trained stably, leading to generate more consistent multi-instruments symbolic music with high quality.

References

[1] Van Der Merwe, Andries, and Walter Schulze, “Music generation with Markov models,” IEEE MultiMedia, vol. 18, no. 3, pp. 78-85, 2011.

[2] Hadjeres, Gaëtan, Frank Nielsen, and François Pachet, “GLSR-VAE: Geodesic latent space regularization for variational autoencoder architectures,” Computational Intelligence (SSCI), 2017 IEEE Symposium Series on. IEEE, 2017.

[3] Eck, Douglas, and Juergen Schmidhuber, “A first look at music composition using lstm recurrent neural networks,” Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale 103, 2002.

[4] Sturm, B. L., Santos, J. F., Ben-Tal, O., & Korshunova, I., “Music transcription modelling and composition using deep learning,” in arXiv preprint arXiv:1604.08723, 2016.

[5] Jaques, N., Gu, S., Turner, R. E., & Eck, D., “Tuning recurrent neural networks with reinforcement learning,” in arXiv preprint arXiv:1611.02796, 2017.

[6] I. J. Goodfellow, J. Pouget-Abadie, and M. Mirza, “Generative adversarial nets,” Advances in Neural Information Processing Systems, pp. 2672–2680, 2014.

[7] Yang, Li-Chia, Szu-Yu Chou, and Yi-Hsuan Yang, “MidiNet: A convolutional generative adversarial network for symbolic-domain music generation,” International Society for Music Information Retrieval Conference, pp. 324–331, 2017.

[8] Mogren, Olof, “C-RNN-GAN: Continuous recurrent neural networks with adversarial training,” Advances in Neural Information Processing Systems, 2016.

[9] Yu, L., Zhang, W., Wang, J., & Yu, Y., “SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient,” The Association for the Advance of Artificial Intelligence, pp. 2852-2858, 2017.

[10] Guimaraes, G. L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C., & Aspuru-Guzik, A., "Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models,” in arXiv preprint arXiv:1705.10843, 2017.

[11] Dong, H. W., Hsiao, W. Y., Yang, L. C., & Yang, Y. H., “MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment,” The Association for the Advance of Artificial Intelligence, 2018.

[12] Dong, Hao-Wen, and Yi-Hsuan Yang, “Convolutional Generative Adversarial Networks with Binary Neurons for Polyphonic Music Generation,” International Society for Music Information Retrieval Conference, pp. 190-196, 2018.

[13] Zhang, H., Goodfellow, I., Metaxas, D., & Odena, A., “Self-Attention Generative Adversarial Networks,” in arXiv preprint arXiv:1805.08318, 2018.

[14] Luo, Ping, Jiamin Ren, and Zhanglin Peng, “Differentiable learning-to-normalize via switchable normalization,” Computer Vision and Pattern Recognition, 2018.

[15] Ioffe, Sergey, and Christian Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in arXiv preprint arXiv:1502.03167, 2015.

[16] Brunner, G., Wang, Y., Wattenhofer, R., & Zhao, S., “Symbolic music genre transfer with cyclegan,” International Conference on Tools with Artificial Intelligence, pp. 786-793, 2018.

[17] Brunner, G., Konrad, A., Wang, Y., & Wattenhofer, R., “MIDI-VAE: Modeling dynamics and instrumentation of music with applications to style transfer,” International Society for Music Information Retrieval Conference, pp. 747-754, 2018.

[18] Vedaldi, Victor Lempitsky Dmitry Ulyanov Andrea, “Instance Normalization: The Missing Ingredient for Fast Stylization,” Computer Vision and Pattern Recognition, 2016.

[19] Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton, “Layer normalization,” in arXiv preprint arXiv:1607.06450, 2016.

[20] He, K., Zhang, X., Ren, S., & Sun, J., “Identity mappings in deep residual networks,” European conference on computer vision. Springer, Cham, pp. 630-645, 2016.

[21] Raffel, Colin., “Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching,” Columbia University, 2016.

[22] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere, “The Million Song Dataset,” International Society for Music Information Retrieval Conference, pp. 591–596, 2011.

[23] Diederik P. Kingma and Jimmy Ba., “Adam: A method for stochastic optimization,” International Conference for Learning Representations, 2014.