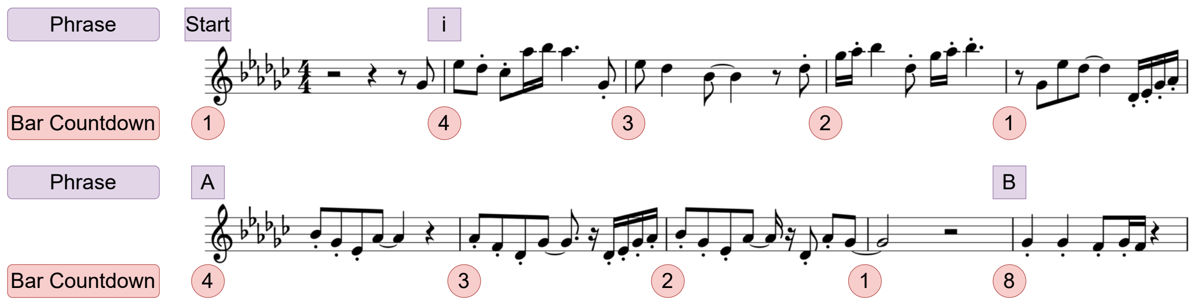


Figure 2: PHRASE and BAR COUNTDOWN events in the staff notation. This is the preprocessed melody part of '001.wav' in POP909 [17].

score. PHRASE (Start) represents one bar before the piece begins. This bar exists in all pieces to consider an anacrusis (or auftakt). Pieces that begin with an anacrusis have some notes in this bar, while those without an anacrusis promptly move to the next bar. PHRASE (End) is placed at the end of the event-token sequence and represents the end of the piece.

The second event, BAR COUNTDOWN, indicates the number of bars remaining in a phrase. If four bars remain, it is expressed as BAR COUNTDOWN (4), and the number of bars is counted for each bar.

These two events are expected to allow the model to know which phrase and where it is generating and to adjust the generation toward the turn of the phrase and the end of the piece. The correspondence between the two events and the musical score is shown in Figure 2. In REMI, the PHRASE and BAR COUNTDOWN events are placed just after the BAR event, which represents a bar line. In CP, these two events are placed along all the time steps. We slightly modified the original settings of CP in the CP Transformer [7] to decompose the METRICAL family into BAR and POS families. The BAR family represents a bar line instead of the BAR event, and the POS family groups the BEAT and CHORD events. In both REMI and CP, performance-related events, NOTE VELOCITY and TEMPO, were not used to reduce the burden of learning. We refer to these extended REMI and CP as **REMI + Ph&BC** and **CP + Ph&BC**, respectively (Ph and BC represent PHRASE and BAR COUNTDOWN, respectively). A list of events used in each representation is shown in Table 1, and an example of each event-token sequence is shown in Figure 3.

3.3 Reflection of User Input

We propose an autoregressive generation method to reflect user input regarding phrase lengths when generating pieces. As shown in Figure 1, event tokens are first predicted using the trained neural sequence model. Before using the predicted event-token sequence as the model input, the PHRASE and BAR COUNTDOWN events are added to the appropriate places based on the user input. This method is expected to enable the input of the phrase lengths that do not exist in the training data because they are present in BAR COUNTDOWN events.

REMI + Ph&BC	CP + Ph&BC	
Event	**Event**	**Family**
Note On Note Duration	Note On Note Duration	Note
Bar	-	Bar
Beat Chord	Beat Chord	Pos
Phrase **Bar Countdown**	**Phrase** **Bar Countdown**	[All]
-	Conti	[All]

Table 1: Events in REMI + Ph&BC and CP + Ph&BC.

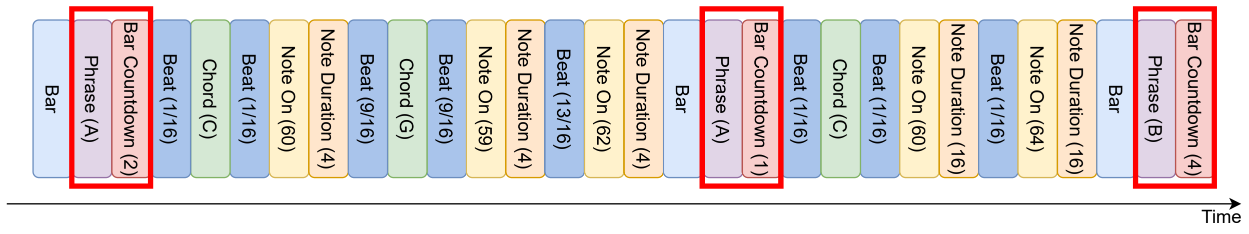
3.4 Pipeline

The same Transformer-based model was used as in the Pop Music Transformer [6] and the CP Transformer [7]. During the training stage, the MIDI file, chord annotation, and phrase annotation of each piece in the dataset are converted into the REMI + Ph&BC or CP + Ph&BC event-token sequence. The model is trained to predict the next event tokens from the input event-token sequence. The pipeline during the generation stage is illustrated in Figure 1. First, BAR, PHRASE (Start), and BAR COUNTDOWN (1) are input to the model to start the generation. Then, as described in Section 3.3, the next event tokens are predicted by the model, and after adding the PHRASE and BAR COUNTDOWN events based on user input, the event-token sequence is again entered into the model. By repeating this process to predict the event tokens sequentially, a piece is generated probabilistically. Once the model has generated the designated number of bars, it adds PHRASE (End) and terminates the generation.

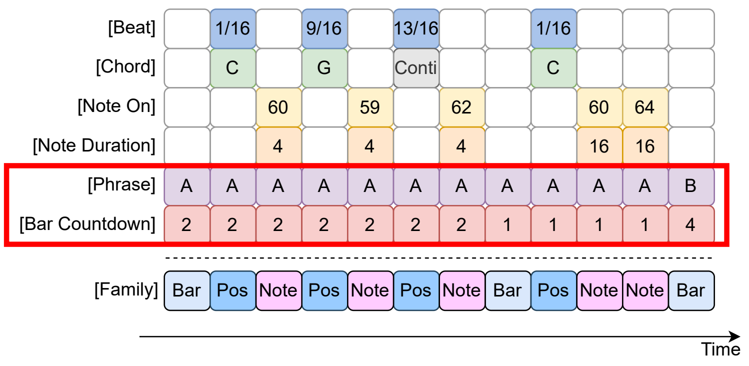
4. EXPERIMENTS

4.1 Dataset

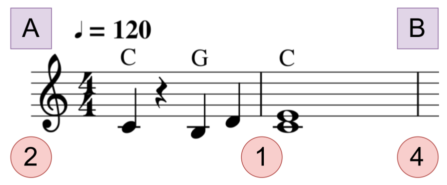
In this study, POP909 [17] is used as the MIDI dataset. The dataset consists of 909 pieces that are piano arrangements of pop songs by professional musicians and is divided into three parts: vocal melody, secondary melody or lead instrument melody, and piano accompaniment. We also used algorithmic chord annotations included in POP909 and the



(a) REMI + Ph&BC



(b) CP + Ph&BC



(c) Musical score

Figure 3: Examples of REMI + Ph&BC (a) and REMI + Ph&BC (b) event-token sequences and the corresponding staff notation (c).

human phrase annotations provided by Dai et al. [18].

After selecting pieces with 4/4 time signatures and excluding those whose downbeats were not aligned with the bar lines, 763 MIDI files were obtained. As a preprocessing step, we merged the three parts and quantized each piece into a 16th-note grid. Next, we shifted all pieces so that the first complete bar was the second to consider the pieces with an anacrusis, as described in Section 3.2. Furthermore, as the number of the intro and outro phrases was smaller than that of the other phrases, we extracted the first and last parts of the pieces and performed data augmentation by transforming their keys in the range of -3 to $+3$. We also modified the chord annotations. All flats on the root note were changed to sharps, and the chord types were limited to the following six types: maj, min, dim, aug, sus4, and sus2.

4.2 Overview of Evaluation Methods

In this study, the length of each phrase and the entire piece was evaluated. In the evaluation of each phrase length, we determined whether each phrase had a designated length by locating the boundary at which the phrase changed. The controllability of the overall length was determined based on whether it ended naturally or abruptly.

For an objective evaluation, methods that can automatically detect phrase boundaries and calculate a naturalness score for the end of a piece are required; however, no suitable methods are available (details are discussed in Section 4.6). In this study, we did not conduct an objective evaluation; rather, we conducted a subjective evaluation.

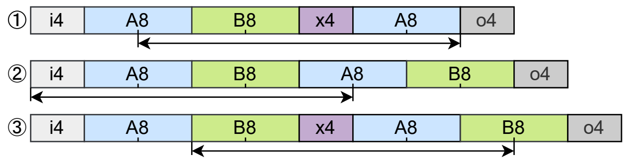
For the subjective evaluation, we administered two listening tests: one to divide a piece into several phrases and the other to evaluate the naturalness of the end. The details are provided in Section 4.4.

4.3 Comparative Methods

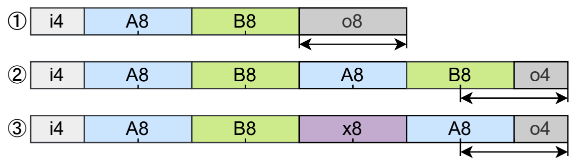
First, we compared our REMI + Ph&BC and CP + Ph&BC with the existing music representations, REMI [6] and CP [7]: **REMI** and **CP**. Since these cannot control the phrase lengths, we evaluated them only for closure. In these methods, we used the events and families shown in Table 1, excluding PHRASE and BAR COUNTDOWN, for comparison under the same conditions. In REMI, generation cannot be terminated by the model; instead, the model is forced to terminate when the number of generated bars reaches the target number. In CP, the model can naturally end a piece, although it cannot control the length of the piece. The final parts of the generated pieces were used for the evaluation.

Next, for the ablation studies, we compared REMI + Ph&BC and CP + Ph&BC with methods with a lower number of events. First, we compared REMI + Ph&BC with **REMI + Ph$_{\text{fewer}}$&BC**, a method that places PHRASE events only at the beginning of phrases. Note that in CP, the number of time steps does not change even if the number of PHRASE events is reduced, so CP + Ph$_{\text{fewer}}$&BC was not evaluated. We also compared our method with methods that used only one of the two events: **REMI + BC**, **REMI + Ph**, **CP + BC**, and **CP + Ph**.

The pieces in the POP909 dataset were also evaluated: **POP909**. Additionally, we intentionally created pieces that ended abruptly by cutting them off in the middle: **POP909$_{\text{cut}}$**. This method was used only when evaluating the closure.



(a) Test 1



(b) Test 2

Figure 4: Inputs of the generated pieces used for the evaluation. One division represents four bars. The segments indicated by arrows were extracted and used.

Method	Test 1	Test 2
POP909	0.778	(3.67)
POP909$_{\text{cut}}$		1.29
REMI + Ph&BC (ours)	**0.633**	**3.00**
REMI + Ph$_{\text{fewer}}$&BC	0.550	2.34
REMI + BC	0.400	1.92
REMI + Ph	0.356	1.27
CP + Ph&BC (ours)	**0.583**	**3.28**
CP + BC	0.461	2.25
CP + Ph	0.306	1.78
REMI		1.35
CP		(3.17)

Table 2: Average percentage of correct answers for phrase boundaries in Test 1 and the average score of the four-grade evaluation of the closure in Test 2. For both tests, higher scores indicate better performance. The bracketed score in Test 2 indicates that it was evaluated with pieces that were not of the designated length.

4.4 Subjective Evaluation

We conducted the following two online listening tests using Amazon Mechanical Turk:

Test 1 Phrase Boundary Detection
We investigated whether each phrase had a designated length or not. The participants listened to a piece while looking at the score and divided it into phrases. They were told in advance how many phrases there were, and they were asked to identify the phrase boundaries.

Test 2 4-Grade Evaluation of Closure
We examined whether the pieces ended naturally or abruptly. The subjects listened to a piece and answered whether the closure was natural or abrupt on a 4-point Likert scale: "Natural," "Somewhat natural," "Somewhat abrupt," and "Abrupt."

Since the majority of phrases in POP909 are four and eight bars in length [12], three inputs, consisting of four- and eight-bar phrases, were used for generation. The concrete inputs are shown in Figure 4. To reduce the burden on the subjects, in Test 1, 24 bars from the middle of a piece containing four phrases, and in Test 2, eight bars from the end were extracted and used for the evaluation. We generated 50 pieces per input and randomly selected two pieces, i.e., six pieces (three inputs $\times$ two pieces) were evaluated for each method.

First, qualification tests were conducted using the dataset to select those who understood each task and performed well, i.e., conformed to the dataset. In Test 1, 24 subjects correctly identified at least six of the nine phrase boundaries for three POP909 pieces, 13 of whom had more than one year of musical experience. In Test 2, 29 subjects answered "Natural" or "Somewhat natural" for two POP909 pieces and "Abrupt" or "Somewhat abrupt" for two POP909$_{\text{cut}}$ pieces, 12 of whom had more than one year of experience.

We then tested eight methods (except REMI, CP, and POP909$_{\text{cut}}$) in Test 1 and all 11 methods in Test 2. Each test was performed 60 times, and one piece per method was evaluated per test. In Test 1, the score was based on the percentage of correct answers, whereas in Test 2, "Natural" was scored as 4 points and "Abrupt" as 1 point.

4.5 Results

The average percentage of correct answers for phrase boundaries in Test 1 and the average score of the four-grade evaluation of the closure in Test 2 are listed in Table 2. In addition, the one-tailed t-test scores comparing our two methods to the other methods are shown in Table 3.

In Test 1, the scores of REMI + Ph&BC and CP + Ph&BC were much higher than 0.130 ($= 3/23$), the score when the phrase boundaries were answered at random. Compared with POP909, the scores were significantly lower (Table 3). It is suggested that our methods are effective in controlling the phrase lengths. When compared to methods used in the ablation studies, our methods scored significantly higher than methods with one of the two events. This indicates that both PHRASE and BAR COUNTDOWN events are required to control the phrase lengths. No significant differences were found between REMI + Ph&BC and REMI + Ph$_{\text{fewer}}$&BC. Thus, to control the phrase lengths, the number of events can be reduced by placing the PHRASE event only at the beginning of the phrase.

In Test 2, our scores exceeded 2.5, which was in the middle of the score range. They were significantly lower than the POP909 score but significantly higher than the POP909$_{\text{cut}}$ score (Table 3). Furthermore, they did not differ from the CP score, where the generated pieces ended naturally. Therefore, it can be said that our methods can end a piece naturally at a designated length. When compared to the methods used in the ablation studies, our methods scored significantly higher than all other methods. This indicates that both PHRASE and BAR COUNTDOWN events are required to control the length of the piece. These results

Method	Test 1		Test 2	
POP909	$\mu < \mu_m{}^{**}$	$(p = 0.0022)$	$\mu < \mu_m{}^{***}$	$(p = 3.2 \times 10^{-5})$
POP909$_{\text{cut}}$			$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 2.0 \times 10^{-17})}$
REMI + Ph$_{\text{fewer}}$&BC	$\mu > \mu_m$	$(p = 0.082)$	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 7.6 \times 10^{-4})}$
REMI + BC	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 1.1 \times 10^{-4})}$	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 2.5 \times 10^{-7})}$
REMI + Ph	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 7.4 \times 10^{-7})}$	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 1.1 \times 10^{-17})}$
REMI			$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 6.9 \times 10^{-16})}$
CP			$\mu < \mu_m$	$(p = 0.18)$

(a) Results of the t-test between REMI + Ph&BC and other methods.

Method	Test 1		Test 2	
POP909	$\mu < \mu_m{}^{***}$	$(p = 1.6 \times 10^{-4})$	$\mu < \mu_m{}^{***}$	$(p = 5.8 \times 10^{-4})$
POP909$_{\text{cut}}$			$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 1.0 \times 10^{-31})}$
CP + BC	$\boldsymbol{\mu > \mu_m{}^{*}}$	$\boldsymbol{(p = 0.024)}$	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 3.3 \times 10^{-9})}$
CP + Ph	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 1.6 \times 10^{-6})}$	$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 3.2 \times 10^{-17})}$
REMI			$\boldsymbol{\mu > \mu_m{}^{***}}$	$\boldsymbol{(p = 1.6 \times 10^{-27})}$
CP			$\mu > \mu_m$	$(p = 0.22)$

(b) Results of the t-test between CP + Ph&BC and other methods.

Table 3: One-tailed t-test scores comparing our two methods to the other methods in Tests 1 and 2. μ and μ_m denote the average score of our method and each of the other methods, respectively. ${}^{*}p < .05, {}^{**}p < .01, {}^{***}p < .001$. A p-value less than 0.05 was considered statistically significant.

also highlight the importance of placing the PHRASE event in every bar (REMI + Ph&BC), not just at the beginning of the phrase (REMI + Ph$_{\text{fewer}}$&BC).

4.6 Discussion

Comparing the method that uses only the PHRASE event and the one that uses only the BAR COUNTDOWN event, the BAR-COUNTDOWN-only method scored higher, regardless of the test or representation (Table 2). This means that the BAR COUNTDOWN event was more effective in controlling the length of each phrase and the entire piece. This is consistent with the role of the BAR COUNTDOWN event in teaching the model the number of bars until the end of the phrase. The reasons why adding the PHRASE event to BAR-COUNTDOWN-only method would improve scores are inferred as follows. In Test 1, the PHRASE event indicates that the phrase has changed and may play a role in making the boundaries of the phrase more distinct. In Test 2, the event can tell the model that the phrase being generated is the outro phrase, and the piece is almost over.

A two-tailed t-test was performed to determine whether there was a significant difference between REMI + Ph&BC and CP + Ph&BC. The results showed that there was no significant difference between these methods, with $p = 0.42$ for Test 1 and $p = 0.11$ for Test 2. We can say that both representations achieve equally good results. These two events could potentially be used for new event-based representations derived from REMI and CP. In addition, although the Transformer was used as the model in this study, it is expected to be widely applied to sequence models that perform autoregressive generation.

As mentioned in Section 4.2, there are no objective evaluation metrics suitable for this study; therefore, we did not conduct an objective evaluation but only a careful subjective evaluation. Although the fitness scape plot [19,20] has been used in studies focusing on the generation of music structures [14, 16], it cannot be used for phrases that do not necessarily repeat, as in this study. Although several algorithms have been proposed to determine phrase boundaries [21], they cannot be adopted because of the low correctness rate when applied to POP909 pieces. The development of phrase-segmentation research and the establishment of objective evaluation methods are required.

One limitation of this study is that it was not possible to create repetitions by designating the same phrase labels in the input. For example, if the input is "A4 B4 A4," the two phrases A are completely different. A possible reason is that the model cannot refer to the next phrase label; therefore, the piece is not connected to the beginning of the next phrase, which was previously defined. A mechanism for making such distant phrases with the same label alike is necessary and is an issue for the future.

In addition, the following points need to be addressed in future studies: (1) combining our methods with music theory to achieve more natural pieces, especially in terms of phrase transition and closure. (2) evaluating length diversity because only a few types of lengths were used because of the convenience of the evaluation. (3) controlling other phrase attributes such as emotions.

5. CONCLUSION

In this study, we proposed a method to control the length of each phrase and the entire piece using a sequential generation policy. In this method, two events are added to the existing event-based music representations: the PHRASE event, which indicates the phrase to which the bar belongs, and the BAR COUNTDOWN event, which indicates the number of bars remaining in the phrase. To reflect the user input, an autoregressive generative model is used that adds these two events based on the user input to the previously generated event-token sequence and uses it as input for the next time step. Subjective listening tests indicated that adding two events effectively controlled the length of each phrase and the entire piece.

6. ACKNOWLEDGEMENT

This work was partially supported by JST AIP Acceleration Research JPMJCR20U3, Moonshot R&D Grant Number JPMJPS2011, CREST Grant Number JPMJCR2015, JSPS KAKENHI Grant Number JP19H01115, and Basic Research Grant (Super AI) of Institute for AI and Beyond of the University of Tokyo.

7. REFERENCES

[1] L. A. Hiller Jr and L. M. Isaacson, "Musical Composition with a High Speed Digital Computer," in *Audio Engineering Society Convention 9*. Audio Engineering Society, 1957.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *NeurIPS*, 2017.

[3] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music Transformer: Generating Music with Long-Term Structure," in *ICLR*, 2019.

[4] C. Donahue, H. H. Mao, Y. E. Li, G. W. Cottrell, and J. McAuley, "LakhNES: Improving multi-instrumental music generation with cross-domain pre-training," in *ISMIR*, 2019.

[5] C. Payne, "MuseNet," *OpenAI Blog*, 2019.

[6] Y.-S. Huang and Y.-H. Yang, "Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions," in *ACM Multimedia*, 2020.

[7] W.-Y. Hsiao, J.-Y. Liu, Y.-C. Yeh, and Y.-H. Yang, "Compound Word Transformer: Learning to Compose Full-Song Music over Dynamic Directed Hypergraphs," in *AAAI*, 2021.

[8] S. Oore, I. Simon, S. Dieleman, D. Eck, and K. Simonyan, "This time with feeling: learning expressive musical performance," *Neural Computing and Applications*, 2018.

[9] Y. Zhou, W. Chu, S. Young, and X. Chen, "BandNet: A Neural Network-based, Multi-Instrument Beatles-Style MIDI Music Composition Machine," in *ISMIR*, 2019.

[10] S. Dai, X. Ma, Y. Wang, and R. B. Dannenberg, "Personalized Popular Music Generation Using Imitation and Structure," *arXiv preprint arXiv:2105.04709*, 2021.

[11] S.-L. Wu and Y.-H. Yang, "MuseMorphose: Full-Song and Fine-Grained Music Style Transfer with One Transformer VAE," *arXiv preprint arXiv:2105.04090*, 2021.

[12] J. Zhao and G. Xia, "AccoMontage: Accompaniment Arrangement via Phrase Selection and Style Transfer," in *ISMIR*, 2021.

[13] S. Dai, Z. Jin, C. Gomes, and R. B. Dannenberg, "Controllable deep melody generation via hierarchical music structure representation," in *ISMIR*, 2021.

[14] X. Zhang, J. Zhang, Y. Qiu, L. Wang, and J. Zhou, "Structure-Enhanced Pop Music Generation via Harmony-Aware Learning," *arXiv preprint arXiv:2109.06441*, 2021.

[15] J. Ens and P. Pasquier, "MMM: Exploring Conditional Multi-Track Music Generation with the Transformer," *arXiv preprint arXiv:2008.06048*, 2020.

[16] S.-L. Wu and Y.-H. Yang, "The Jazz Transformer on the Front Line: Exploring the Shortcomings of AI-composed Music through Quantitative Measures," in *ISMIR*, 2020.

[17] Z. Wang, K. Chen, J. Jiang, Y. Zhang, M. Xu, S. Dai, G. Bin, and G. Xia, "POP909: A Pop-song Dataset for Music Arrangement Generation," in *ISMIR*, 2020.

[18] S. Dai, H. Zhang, and R. B. Dannenberg, "Automatic Analysis and Influence of Hierarchical Structure on Melody, Rhythm and Harmony in Popular Music," in *CSMC-MuMe*, 2020.

[19] M. Müller, P. Grosche, and N. Jiang, "A Segment-Based Fitness Measure for Capturing Repetitive Structures of Music Recordings." in *ISMIR*, 2011.

[20] M. Müller and N. Jiang, "A Scape Plot Representation for Visualizing Repetitive Structures of Music Recordings," in *ISMIR*, 2012.

[21] O. Nieto, G. J. Mysore, C. i Wang, J. B. L. Smith, J. Schlüter, T. Grill, and B. McFee, "Audio-Based Music Structure Analysis: Current Trends, Open Challenges, and Applications," *Trans. Int. Soc. Music. Inf. Retr.*, vol. 3, pp. 246–263, 2020.

EXPLOITING PRE-TRAINED FEATURE NETWORKS FOR GENERATIVE ADVERSARIAL NETWORKS IN AUDIO-DOMAIN LOOP GENERATION

Yen-Tung Yeh^{1,2} **Bo-Yu Chen**^{1,2} **Yi-Hsuan Yang**^{1,3}

^{1} Academia Sinica, ^{2} National Taiwan University, ^{3} Taiwan AI Labs

ytsrt66589@gmail.com, bernie40916@gmail.com, yang@citi.sinica.edu.tw

ABSTRACT

While generative adversarial networks (GANs) have been widely used in research on audio generation, the training of a GAN model is known to be unstable, time consuming, and data inefficient. Among the attempts to ameliorate the training process of GANs, the idea of Projected GAN emerges as an effective solution for GAN-based image generation, establishing the state-of-the-art in different image applications. The core idea is to use a pre-trained classifier to constrain the feature space of the discriminator to stabilize and improve GAN training. This paper investigates whether Projected GAN can similarly improve audio generation, by evaluating the performance of a StyleGAN2-based audio-domain loop generation model with and without using a pre-trained feature space in the discriminator. Moreover, we compare the performance of using a general versus domain-specific classifier as the pre-trained audio classifier. With experiments on unconditional one-bar drum loop and synth loop generation, we show that a general audio classifier works better, and that with Projected GAN our loop generation models can converge around 5 times faster without performance degradation.

1. INTRODUCTION

Generative adversarial networks (GANs) [1] are deep generative models that are composed of a *generator* and a *discriminator*. The discriminator can be considered as a classifier (or a "critic") which judges whether its input is a real sample, or a synthetic one created by the generator counterpart; the generator takes as input a random vector (sometimes plus additional conditional inputs [2]) and aims to create a synthetic sample that "fools" (i.e., appears to be realistic to) the discriminator. During the GAN training process, the discriminator and generator are updated in an iterative manner, fixing the parameters of one and updating those of the other each time. After the training converges, the generator can be used to generate original samples, leading to state-of-the-art (SOTA) results in image generation [3–7] and many other domains.

GANs have been extensively applied to music generation as well, including symbolic-domain generation [8–17] and audio-domain generation [18–28]. In particular, GAN-based models represent the SOTA in audio-domain music generation tasks such as single-note generation [18, 26], drum track generation [21], and loop generation [27].

Despite its widespread applications, GANs are notoriously difficult to train [29–33]. This is partly due to the fact that the generator and discriminator have opposite goals by design, with the generator aiming to *maximize* the discriminator loss and the discriminator aiming to *minimize* the same loss in different iterations, making the training dynamics complicated. As the parameters of the discriminator are constantly being updated over the training process, the generator has no single critic to improve its own performance over time. And, while the parameters of the generator and discriminator are typically initialized *randomly*, the GAN training process can be time-consuming. Using a pre-trained feature network as part of the discriminator has been shown to speed up the training process [34–36], but some modifications of the pre-trained feature network is needed to avoid the discriminator from being too strong and causing the gradients of the generator to vanish.

Projected GAN [37] is a new approach that is shown to effectively leverage the benefits of pre-trained feature networks, leading to SOTA results in unconditional image generation [7]. Projected GAN adds two *random projection* modules after the pre-trained feature networks, named "cross-channel mixing" (CCM) and "cross-scale mixing" (CSM), to prevent the discriminator from focusing only on a subset of features and thereby avoid mode collapse. They not only reduce the time of GAN training, but also improve the quality of the generated images.

This paper studies whether Projected GAN can also improve audio generation, by incorporating it to unconditional audio-domain loop generation [27] as a case study. We note that there are well-studied pre-trained feature networks in the image domain [34–36]. There are also studies on the choice of pre-trained network for retrieval and analysis of musical audio [38] and soundtracks [39]. However, we are less sure which pre-trained feature networks to use in the context of musical audio generation. Therefore, besides pioneering the use of Projected GAN for audio generation, an interesting aspect of our research is that we in-

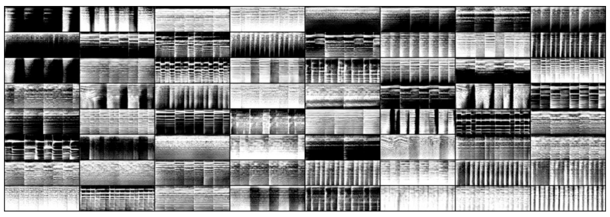


Figure 1. Mel-spectrograms of examples of synth loops generated by Projected GAN with a general classifier.

vestigate different types of pre-trained feature networks for Projected GAN-based musical audio generation, including general classifier and domain-specific classifiers.

Specifically, for the **general** classifier, we use the pre-trained VGGish feature networks trained on YouTube-100M [39], a huge collection of sounds and musical audio from YouTube. For **domain-specific** classifiers, we use short-chuck convolutional neural network (SCNN)-based models [40], which have led to SOTA accuracy across multiple music auto-tagging datasets. We use two SCNNs, one trained on the MagnaTagATune (MTAT) dataset [41] for tagging music of a wide range of genres, and the other trained on a collection of loops for genre classification of loops, which are domain-wise even closer to our generation task. We elaborate on the datasets and classifiers in Section 3, and then the usage of the classifiers as the pre-trained feature networks for loop generation in Section 4.

We report objective and subjective evaluations in Sections 5 and 6 studying the influence of different pre-trained feature networks for drum loop generation and synth loop generation following the approach of Projected GAN, finding that the use of a general classifier works consistently better. Moreover, we find slightly better result can be obtained if we “fuse” general and domain-specific pre-trained feature networks in our model. Objective and subjective evaluations both demonstrate that Projected GAN improves training speed and the final generation quality.

We share our code at `https://github.com/Arthurddd/pjloop-gan`, and examples of the generated loops at `https://arthurddd.github.io/PjLoopGAN/`. Illustrations of the Mel-spectrograms of the generated synth loops and drum loops can be found respectively in Figure 1 and later on in Figure 3.

2. BACKGROUND

Related Work. GANs have garnered great interest in recent years, leading to models that generate high-fidelity audio waveforms. WaveGAN [19] pioneers the use of GAN for audio; it can synthesize one-second raw audio waveforms of speech and music with coherence. Follow-up research applies GAN to other audio synthesis tasks. For example, MelGAN [42] and Parallel WaveGAN [43] use GAN to build neural vocoders that can reconstruct a waveform from the corresponding Mel-spectrogram. GAN-TTS [44] applies GAN to convert text to natural human speech. UNAGAN [23] uses GAN to synthesize singing voice.

GAN has also been used to synthesize audio samples of musical materials, including percussive and harmony ones, that can be used in music production. GANSynth [18] can synthesize harmony music notes with diverse timbre and controllable pitches. DrumGAN [24] can synthesize one-shot percussion sounds, allowing for the use of condition perceptual features to intuitively control the timbre of the percussion sounds. StyleWaveGAN [45] improves the audio quality and inference time of DrumGAN. However, the aforementioned models deal with only single notes or one-shot samples instead of longer phrases such as musical loops, limiting their applicability to creating loop-based music. In view of this need, Hung *et al.* [27] study the task of one-bar drum loop generation and benchmark the performance of UNAGAN [23], StyleGAN [4] and StyleGAN2 [5] for this task over the FreeSound Loop dataset [46], showing that the model based on StyleGAN2 [5] performs the best. However, Hung *et al.* [27] do not consider the training efficiency of their models. Moreover, while they focus on drum loops only, we consider also the generation of synth loops to encompass not only percussive but also harmonic musical materials in our work.

Specific to the GAN training technique, there are three main approaches to improve the discriminator. First, modifying the *architecture* of the discriminator to improve the discriminator's ability [47–49]. Second, assembling *multiple discriminators* to capture more comprehensive features, an approach that as been widely investigated in the audio domain for building the vocoder [42, 50–52]. The third approach introduces a *pre-trained feature network* to the discriminator to avoid learning its parameters completely from scratch, which helps speed up the convergence time and prevent model overfitting. Zhao *et al.* [34] use the pre-trained network in the large-scale dataset, adapt it to a small dataset, and propose adaptive filter modulation to deal with domain shift. Grigoryev *et al.* [35] and Kumari *et al.* [36] both find that pre-trained network selection can largely influence performance and propose a recipe to choose a suitable pre-trained network for a particular image-domain task. Projected GAN [37] is a very recent idea that combines the aforementioned approaches, using multiple discriminators and employing pre-trained feature networks simultaneously to improve the training stability and efficiency of the GAN. However, to our best knowledge, adapting Projected GAN to the audio-domain generation is still unexplored, for either speech or music.

Projected GAN. Projected GAN [37] introduces a set of $L = 4$ feature projectors $\{P_l, l = 1, \ldots, L\}$, each maps either real samples $\mathbf{x}$ or fake samples $G(\mathbf{z})$ to a fixed pre-trained feature space. It aims to minimize the following objective function to match the data distribution in the feature space, instead of matching data distributions directly:

$$\begin{aligned}\min_{G}\max_{\{D_l\}} \sum\nolimits_{l=1}^{L} \exp\big(&E_x[\log D_l(P_l(\mathbf{x}))] \\ &+E_z[\log(1 - D_l(P_l(G(\mathbf{z}))))]\big)\,,\end{aligned} \tag{1}$$

where $\{D_l, l = 1, \ldots, L\}$ denotes the set of independent discriminators operating on different feature projections,