Anchored Cyclic Generation: A Novel Paradigm for Long-Sequence Symbolic Music Generation

Anonymous submission

Abstract

Generating controllable long-sequence music with complete compositional structure is a fundamental challenge in symbolic music generation, where existing methods primarily rely on autoregressive models. Constrained by the inherent severe error accumulation problem of autoregressive models, existing methods exhibit poor performance in terms of music quality and structural integrity when generating long-sequence symbolic music. In this paper, we propose the Anchored Cyclic Generation (ACG) paradigm, which relies on anchor features from already identified music to guide subsequent generation during the autoregressive process, effectively mitigating error accumulation in autoregressive methods. Based on the ACG paradigm, we further propose the Hierarchical Anchored Cyclic Generation (Hi-ACG) framework, which employs a systematic global-to-local generation strategy and is highly compatible with our specifically designed piano token, an efficient musical representation. The experimental results demonstrate that compared to traditional autoregressive models, the ACG paradigm achieves reduces cosine distance by an average of 34.7% between predicted feature vectors and ground-truth semantic vectors. In long-sequence symbolic music generation tasks, the Hi-ACG framework significantly outperforms existing mainstream methods in both subjective and objective evaluations. Furthermore, the framework exhibits excellent task generalization capabilities, achieving superior performance in related tasks such as music completion.

Introduction

Symbolic music generation, as a core branch of music generation, produces discrete musical representations with structured, interpretable characteristics (Briot, Hadjeres, and Pachet 2017). In recent years, with the rapid development of deep learning technologies, researchers have worked to improve the musical quality and expressive capability of symbolic music generation, and modeling long-sequence symbolic music has become a new primary challenge.

Long-sequence symbolic music generation requires not only maintaining local musical coherence but also preserving structural integrity and stylistic consistency at the global level. Among existing approaches for long-sequence symbolic music modeling, autoregressive generative models constitute the mainstream methodology. These methods achieve sequence generation by predicting subsequent musical segments based on historical content, but exhibit significant limitations in long-sequence modeling. Early autoregressive approaches primarily employed RNN or LSTM architectures, which commonly exhibit degraded musical quality and stylistic drift when processing longer sequences, making it difficult to maintain long-term musical consistency. Attention-based methods, such as Transformer and GPT models, typically combined with musical representation schemes like MIDI event encoding (Oore et al. 2020) or ABC notation, demonstrate excellent performance in short-sequence generation. However, as sequence length increases, these methods face two core challenges: First, computational complexity grows exponentially (Child et al. 2019), leading to dramatically reduced training and inference efficiency; second, error accumulation effects become increasingly severe, making it difficult to guarantee generation quality. Furthermore, existing methods struggle to achieve an ideal balance between precise temporal control and structural integrity preservation in music. Another category of symbolic music generation methods employs diffusion models (Mittal et al. 2021) to decode music from latent space vectors, but these approaches are typically limited by the expressive capacity of model outputs, making it difficult to generate complete long-sequence music within short time periods.

Addressing these challenges, we propose the Anchored Cyclic Generation (ACG), a stable paradigm for long-sequence symbolic music generation. The core innovation of this paradigm lies in introducing determined musical content as anchors in each generation cycle to recalibrate the generation process, thereby effectively correcting potential error accumulation problems and achieving smooth transitions between musical segments. ACG can ensure local musical quality while maintaining structural integrity of long-sequence music. Additionally, compared to traditional methods, our approach demonstrates significant advantages in time complexity, substantially improving computational efficiency. Based on the ACG paradigm, we further propose a Hierarchical Anchored Cyclic Generation (Hi-ACG) framework to accomplish complete long-sequence symbolic music generation tasks, and design a more effective symbolic music representation scheme to adapt to this framework. The framework adopts a hierarchical music generation strategy, comprising a sketch information prediction loop and a refinement information prediction layer. The sketch information prediction loop is responsible for capturing and predicting high-level semantic features of music, such as modality, harmonic progression, and overall structure, encoding this information as musical thumbnails to provide global guidance for subsequent refinement processes. The refinement information prediction loop focuses on refining global sketch into concrete musical implementations, processing note-level detailed information to ensure local coherence and expressiveness of generated music. Experimental results demonstrate that our proposed Hi-ACG framework can maintain long-term structural and stylistic consistency in music while achieving precise control over generated music duration.

Overall, our contributions are as follows:

- We present the ACG paradigm, which significantly mitigates error accumulation in long-sequence generation tasks such as symbolic music modeling. Our method demonstrates improved time complexity and lower computational costs compared to conventional autoregressive approaches.
- We present Hi-ACG, a hierarchical framework that generates music from global to local levels. It solves structural integrity problems in long sequences, provides precise duration control, and offers high interpretability.
- We propose a Piano Token musical representation—an efficient tokenization method that converts piano roll data into musical tokens. This approach is highly compatible with our Hi-ACG framework while remaining adaptable to other autoregressive symbolic music generation models.
- Based on mathematical analysis and experimental validation, our proposed ACG paradigm and Hi-ACG framework effectively mitigate error accumulation, enhance the generation quality and structural integrity of long-sequence symbolic music, and demonstrate superior performance over existing models in both objective and subjective evaluations.

Related Work

Symbolic Music Generation

Symbolic music generation is a key research direction in deep music generation, aiming to automatically generate discrete musical representations with musicality and interpretability through algorithms. Early approaches were based on rules and statistical modeling, such as Ebcioglu's expert system for four-part chorale harmonization in the style of Bach (Ebcioglu 1988), and the multiple viewpoint system proposed by Conklin and Witten (Conklin and Witten 1995), which laid the foundation for later data-driven methods.

With the advancement of deep learning, neural networks have become the mainstream approach. Recurrent Neural Networks (RNNs) have been widely used to model temporal dependencies in music (Eck and Schmidhuber 2002)(Johnson 2017)(Sturm et al. 2016). MusicVAE (Roberts et al. 2018), based on a VAE model, introduced a hierarchical encoder-decoder structure enabling interpolation and variation of music segments. SeqGAN (Yu et al. 2017) and MuseGAN (Dong et al. 2017) introduced adversarial mechanisms to enhance diversity and style learning. However, these early methods generally struggled with unstable training and degraded generation quality when handling long sequences.

In recent years, emerging architectures such as Transformers (Vaswani et al. 2017) and diffusion models (Ho, Jain, and Abbeel 2020) have significantly advanced the field. RIPO Transformer (Guo, Kang, and Herremans 2023) leveraged biased sinusoidal embeddings and novel attention mechanisms to enhance melody modeling. MusER (Li et al. 2023) introduced decoupled modeling of musical elements and a dual-decoder architecture, enabling emotionally controllable generation. TunesFormer (Wu et al. 2023) combined a two-stage decoder with ABC notation to enable bar-level controlled generation.

Additionally, ChatMusician (Yuan et al. 2024) showed how language models can understand and generate music. It was pretrained on both text and ABC music notation, creating a unified approach for applying large models to music tasks.

Long-Sequence Symbolic Music Modeling

Long-sequence symbolic music modeling is a key challenge in music generation. Traditional autoregressive methods have several problems like error accumulation, high computational complexity, and vanishing gradients.

Transformer-based Methods Transformers are widely used in long-sequence symbolic music generation due to their ability to model long-range dependencies. Music Transformer (Huang et al. 2018) first applied Transformer architecture to symbolic music generation using relative positional encoding. Longformer (Beltagy, Peters, and Cohan 2020) introduced sparse attention to reduce computational cost. Museformer (Yu et al. 2022) proposed structure-aware FC-Attention using bar-level summary tokens and multi-scale attention. Compound Word Transformer (Hsiao et al. 2021a) introduced compound token representation. BPE-Music (Fradet et al. 2023) employed subword modeling to compress token sequences. MuPT (Qu et al. 2024) introduced a scalable pretraining model using ABC notation and multi-track Transformer design. While powerful, Transformers still face challenges in extremely long-sequence generation due to error accumulation and quality degradation.

Diffusion-based Methods Diffusion models offer a non-autoregressive generation approach with strong fidelity. Discrete diffusion models (Plasser, Peter, and Widmer 2023) have been applied to symbolic music generation, demonstrating state-of-the-art sample quality and flexible note-level infilling capabilities. Diff-Music (Nistal et al. 2024) first applied discrete diffusion to MIDI generation. Cascaded-Diff (Wang, Min, and Xia 2024) employed a multi-step diffusion sampling process to generate music progressively from high-level structure to detailed melody. However, diffusion models are computationally expensive

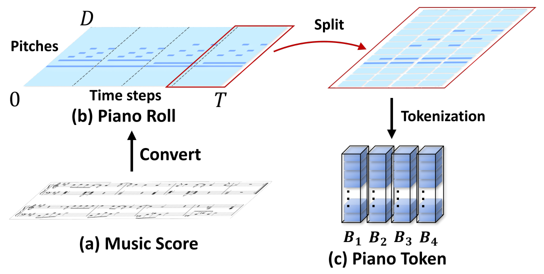

Figure 1: Converting musical scores to piano tokens. (a) Example musical score. (b) Piano roll with T time steps and D pitch dimensions. The red-boxed section demonstrates split details. (c) Piano token representation, where each sub-unit of the split piano roll is tokenized into a piano token. Piano tokens in the same column collectively form a block.

for long sequences due to multi-step sampling, and often struggle with maintaining coherence and duration control.

Hierarchical Music Generation

Hierarchical music generation addresses long-sequence challenges by decomposing generation into multiple levels (e.g., sections, phrases, notes), enhancing structure control and efficiency. Early models like Hierarchical RNN (Zhao, Li, and Lu 2019) used multi-layer recurrent structures to learn temporal patterns. MuseNet (Goren, Nachmani, and Wolf 2021) adopted an implicit hierarchical strategy with gradually increasing context. MMM (Ens and Pasquier 2020) established implicit structural representations through multi-track conditional generation mechanisms. SymphonyNet (Liu et al. 2022) modeled movements, phrases, and notes for symphonic music. Cascaded-Diff (Wang, Min, and Xia 2024) integrated structural language modeling with diffusion techniques to achieve hierarchical music generation from global structure to local details. GraphMuse (Karystinaios and Widmer 2024) modeled hierarchical control through musical graphs. Despite progress, current methods still face issues in inter-level information flow, modeling coordination, and consistency. Particularly in long-sequence generation, achieving efficient transmission and precise duration control across levels remains challenging.

In summary, current symbolic music generation methods face challenges such as error accumulation, low efficiency, and structural inconsistency in long-sequence modeling. We propose an anchored cyclic generation paradigm and hierarchical framework that address these issues through novel generation mechanisms and architectural designs, thereby providing new solutions for long-sequence symbolic music generation.

Method

In this section, we introduce the Piano Token representation, the anchored cyclic generation paradigm, and the hierarchical anchored cyclic generation framework designed based on this paradigm for generating high-quality long-sequence symbolic music.

Piano Token Representation

Common symbolic music representations primarily use MIDI event-based representations, such as REMI(Huang and Yang 2020)or Compound Word representations(Hsiao et al. 2021b), or ABC notation representations. The sequence length of these representation forms exhibits a non-linear relationship with music duration. When music complexity is high and note changes are frequent, extremely long representation sequences are often required. This typically causes severe error accumulation and missing important tokens during music sequence generation.

To address this issue, we design the piano token representation based on piano roll, which is a more efficient music representation method shown in Figure 1. The piano token representation explicitly encodes temporal sequences, where the sequence length L is positively correlated with music duration T, expressed as $L \propto T$. This representation method maps continuous music representations to sparse discrete representations through tokenization while preserving the original spatiotemporal structure. Specifically, we first split the piano roll representation into N patches, where each patch contains $d \times t$ elements from the piano roll, with d representing the number of pitch dimensions covered by a single patch and t denoting the number of time steps covered by a single patch. We then tokenize each patch using a single token to encode its complete content, achieving data compression. Since the piano roll representation is a binary matrix of shape $D \times T$, where D represents the pitch dimension and T denotes the number of time steps. Each element $x_{p,t} \in \{0, 1\}$ indicates whether pitch p is activated at timestep t. The partitioned patches retain this characteristic, so the vocabulary size corresponding to the piano token representation is 2^{d*t} . By adjusting the values of d and t, we can flexibly control the patch size, thereby regulating the vocabulary scale and encoded sequence length to accommodate different application scenarios. After tokenization, the original matrix of size $D \times T$ is converted into a piano token matrix of shape $\frac{D}{d} \times \frac{T}{t}$. The piano token matrix representation serves as the core representational for music and participates in subsequent music generation processes.

We define each column of the piano token matrix as a block B, which contains musical information from t consecutive time steps in the original piano roll representation. Each block contains complete musical fragments within short time intervals, and this block structure also plays a crucial role in subsequent music generation workflows.

Anchored Cyclic Generation Paradigm

Error accumulation is a common problem in autoregressive models, where discrepancies exist between the model's generation and optimal prediction value in each iteration round, and these errors accumulate throughout the iterative process, ultimately leading to severe degradation of the overall generation quality. Error accumulation can be regarded as the primary factor contributing to the degradation of generation quality in long-sequence music generation. To mitigate this issue, we propose the ACG, a novel generation paradigm that can significantly reduce error accumulation during long sequence generation. We draw inspiration

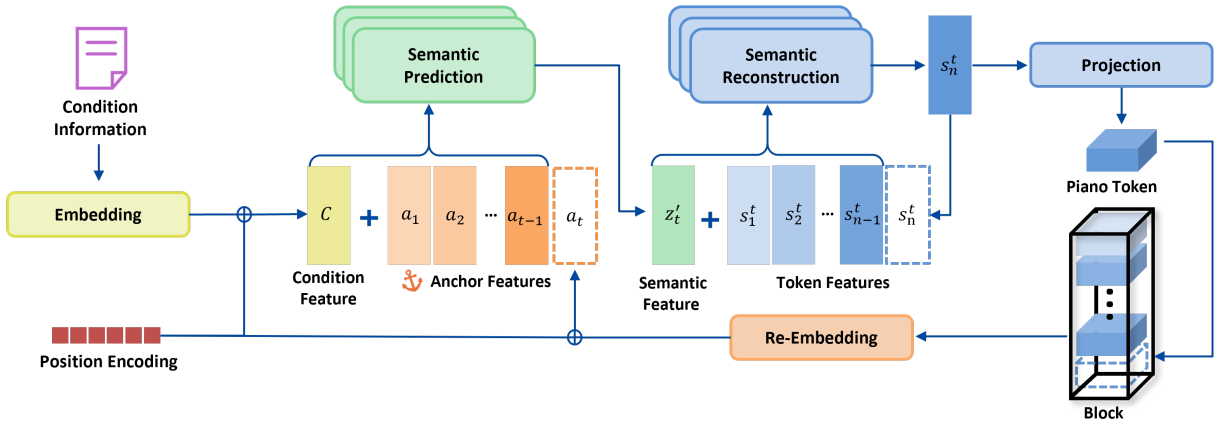

Figure 2: ACG Paradigm Architecture. The embedding layer encodes the conditional information into feature vectors, concatenated with anchor features from existing content (anchor features are absent at the initial time step) and feeds into the semantic prediction model. The semantic prediction model generates semantic features z for the current time step and feeds it to semantic reconstruction model, which autoregressively generate a sequence of features corresponding to piano tokens s through multiple iterations. The generated token feature s concatenates with semantic feature z for iteration. The n piano tokens combine into a block as final output for semantic feature z, then convert into an anchor feature for iterative generation by re-embedding model.

from teacher forcing training methodology: when generating symbolic music sequences, at each iteration, the model predicts features for the next time step t based on determined historical information, which we call anchor features $A_{t-1} = \{a_1, a_2, a_3, ..., a_{t-1}\}$, thereby minimizing the error between the current prediction and the optimal value.

As illustrated in Figure 2, an ACG structure comprises three key components: a semantic prediction model, a semantic reconstruction model, and a specialized re-embedding layer. The semantic prediction model and semantic reconstruction model consist of two cascaded transformer decoder models. The semantic prediction model is responsible for predicting the semantic features z'_t of the current time step t based on input conditions C and anchor features A_{t-1}. It predicts the content of a whole block, which contains token combinations' information. The expression is as follows:

$$z'_t = f_{\text{sem}}(A_{t-1}, C, t; \theta_{\text{sem}})$$

The semantic reconstruction model decodes the semantic feature z'_t and projects it into Piano Token sequence $S^t_{\text{token}} = \{s^t_1, s^t_2, s^t_3, ..., s^t_n\}$ via an additional linear layer, where n denotes the sequence length. The semantic reconstruction process can be expressed as follows:

$$S^t_{\text{token}} = f_{\text{proj}}(f_{\text{rec}}(z'_t; \theta_{\text{rec}}); W_{\text{proj}}, b_{\text{proj}})$$

The re-embedding layer is a neural network composed of multiple fully connected layers, which is responsible for remapping the reconstructed token sequence S^t_{token} back to anchor features a_t for generation in the next iteration.

$$a_t = f_{\text{reemb}}(B_t; \theta_{\text{reemb}})$$

In our design, the three components of the ACG paradigm are jointly trained in an end-to-end manner. It should be noted that in the ACG paradigm, the semantic prediction model does not independently generate all latent vectors A of semantic information before the semantic restoration model sequentially restores them to token representations. In our method, the semantic prediction model first predicts the semantic latent feature z'_t for the current time step t and transmits it to the semantic restoration model and additional projection layer, which then autoregressively decodes a sequence S^t_{token} containing all tokens in the block based on feature z'_t. Each element in sequence S^t_{token} is stacked and rearranged to form the final output block B_t for the current time step. Additionally, we treat the block B_t obtained in each iteration as confirmed historical information and input B_t into a re-embedding layer to transform it back into an anchor semantic feature a_t. The anchor feature a_t for the current time step is concatenated with features from all previous time steps A_{t-1} and fed into the semantic decoder for semantic feature prediction of the next time step. This anchor feature derived from confirmed content can better approximate the optimal feature, guiding the semantic prediction model to achieve more accurate outputs through what we refer to as the anchor mechanism. In the task of generating subsequent musical content given an opening, we extracted 100 samples each of semantic features z' from the ACG paradigm and semantic features z from conventional autoregressive methods, and compared them by computing cosine distances with ground truth, thereby confirming that our hypothesis holds in practice. The result shown in Figure 3. Compared to traditional autoregressive models, the ACG paradigm achieves an average reduction of 34.7% in cosine distance between predicted feature vectors and ground-truth semantic vectors. For the mathematical proof of the effectiveness of the ACG paradigm, please refer to the supplementary materials.

This demonstrates that our proposed ACG paradigm ef-

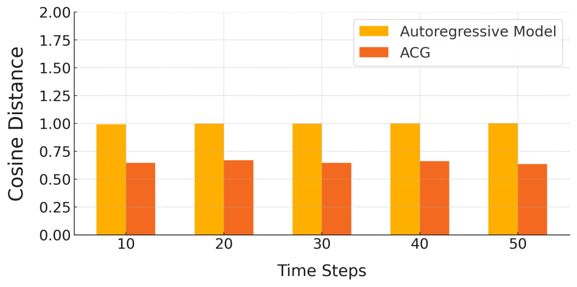

Figure 3: Cosine distances between predicted and ground truth feature vectors for the ACG paradigm and conventional autoregressive models across iterations. The ACG paradigm consistently achieves lower cosine distances compared to conventional autoregressive models.

fectively mitigates the error accumulation inherent in autoregressive models, thereby achieving superior generation performance. In terms of time complexity, ACG also outperforms conventional approaches. The time complexity $O(L^2)$ of conventional autoregressive models scales quadratically with increasing sequence length L, whereas the time complexity of the ACG paradigm is $O(L_{\text{sem}}^2) + L_{\text{sem}} \times O(L_{\text{rec}}^2)$, where L_{sem} represents the length of semantic features sequence Z', and L_{rec} represents the length of token sequence S. The ACG paradigm decomposes autoregressive generation tasks into a two-stage subtask framework, implemented through employing separate semantic prediction and semantic restoration models. The semantic prediction model operates solely at the block level to predict semantic features, while the semantic restoration model focuses exclusively on reconstructing fixed-length token sequences from block features. This task decomposition significantly reduces the computational burden of models in long-sequence autoregressive generation tasks, with the efficiency gains becoming more pronounced as sequence length increases.

Hierarchical Anchored Cyclic Generation Framework

We propose the Hi-ACG framework, built upon the ACG paradigm, to generate high-quality, long-sequence symbolic music with complete structure and precise duration control. This cascaded model architecture embodies a core principle: simulating human compositional cognition through hierarchical music generation that proceeds from global structure to local refinement. This approach mirrors how composers naturally work—first establishing the overall structural framework, then gradually developing specific musical details. By doing so, the framework maintains both global musical coherence and rich local expression, ultimately producing more natural and musically acceptable compositions.

The Hi-ACG framework features two interconnected loops: the Sketch Loop and the Refinement Loop. The Sketch Loop generates high-level structural sketches that establish the compositional backbone. Building on this sketch information, the Refinement Loop focuses on creating rich expressive details, transforming abstract structures into concrete musical content. This clear division of responsibilities allows the framework to optimize music generation at different levels of abstraction, ensuring both structural coherence in long sequences and precise control over duration while enhancing local musical quality.

We propose an approach where the Sketch Loop and Refinement Loop are trained separately with different objectives. We obtain training data for the Sketch Loop by resampling real music data, performing two samplings within each measure to extract each measure's core musical information. This approach preserves the main musical characteristics of each measure while significantly reducing sequence length. After resampling the entire composition, we convert the results to piano token representation for Sketch Loop training. The Refinement Loop uses paired training, utilizing sketches generated by the Sketch Loop paired with corresponding complete musical pieces. We also convert the training data to piano token representation to maintain consistency. During training, the Refinement Loop learns to expand sketch information into complete, detailed musical content, learning to map abstract structures to concrete musical expressions.

During the music generation phase, the framework follows this workflow: First, the Sketch Loop generates a piano token matrix of the complete composition sketch based on conditions such as duration or musical input, providing the overall structural framework. The sketch is then segmented into sequence $B_{\text{sketch}} = \{b_1, b_2, ..., b_t\}$ block-by-block, with each element in sequence B_{sketch} passed individually to the Refinement Loop for processing. The Refinement Loop uses each input block b_t to expand and generate corresponding detailed musical content $B_{\text{refinement}}$. Finally, all refined block sequences are combined in chronological order to form the complete musical work. Through this hierarchical generation strategy, our framework can generate high-quality long-sequence symbolic music with rich detailed expression while ensuring overall structural coherence. For structural control, the global planning of the Sketch Loop ensures that generated music maintains coherent overall structure, avoiding the structural drift commonly seen in long-sequence generation. To ensure quality, the Refinement Loop focuses on optimizing local musical expression, producing music that maintains structural coherence while featuring rich musical details and expressiveness. Moreover, the hierarchical design provides the framework with strong scalability, enabling it to handle music sequences of arbitrary length and making it technically feasible to generate ultra-long musical compositions.

Experiment

To validate the Hi-ACG framework's effectiveness, we conduct comprehensive objective and subjective experiments evaluating the generation quality. This chapter presents the experimental setup and evaluation results.

Experimental Setup

Dataset We train our model on the MuseScore dataset and the POP909 dataset. The MuseScore dataset contains 140,000 two-track piano scores lasting 1-3 minutes. We convert them to piano rolls in multi-hot array format, with a

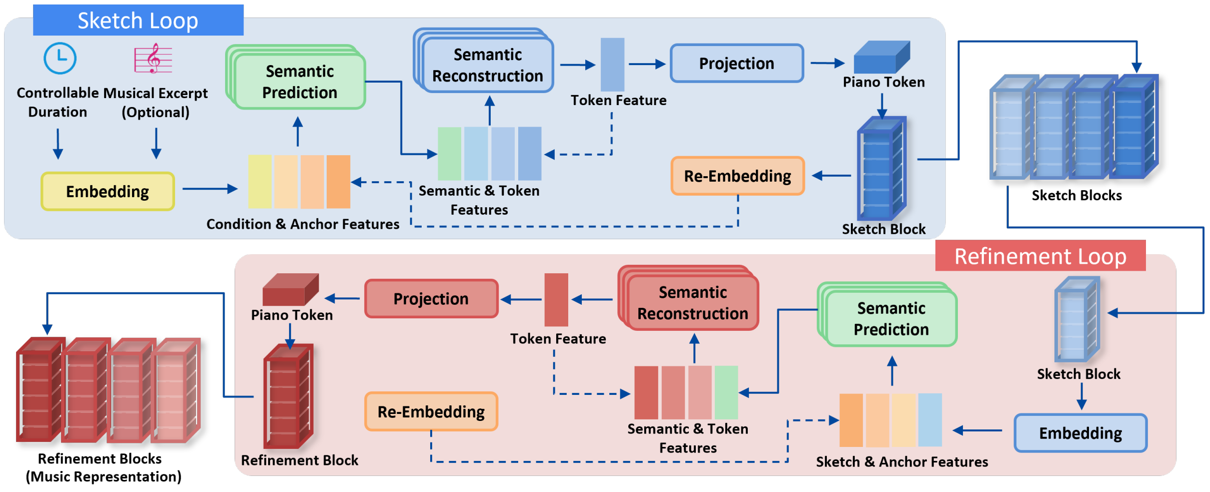

Figure 4: The Hi-ACG framework, which comprises a sketch loop and a refinement loop. The sketch loop takes duration condition and optional musical constraints as input, generating a musical sketch with coarse-grained information for the entire composition using block-by-block processing. The refinement loop then sequentially processes the sketch blocks, transforming each into multiple detailed blocks with rich musical representations.

minimum resolution of 1/16 beat. Each time step preserves 88 pitches corresponding to standard piano keys. We encode the piano rolls into piano tokens to train our model. For POP909, we similarly convert them into piano roll representations with the same temporal resolution, then transform them into piano tokens.

Details During data preprocessing in our experiments, we set d to 2 and t to 4, with each token corresponding to a patch range of 2×4 elements. This results in a piano tokens matrix of size $44 \times (\frac{T}{4})$. For model hyperparameters, in the ACG paradigm, the semantic prediction model contains 12 self-attention layers, the semantic reconstruction model contains 6 self-attention layers, and the re-embedding layer consists of 3 fully connected layers, all models use a hidden dimension of 1024. We trained on the MuseScore data for 30 epochs using 4 NVIDIA RTX 4090 GPUs, then continued fine-tuning our pre-trained model using POP909 data to improve generation performance.

Evaluation

Baseline We selected baseline models from two different architectures for comparison with our model: Transformer-based and diffusion-based approaches. The Transformer-based models include the BPE Transformer model and the Music Transformer model. BPE Transformer introduces the BPE tokenization technique from natural language processing into symbolic music generation. This approach is particularly effective for long-sequence symbolic music generation, while also showing superior performance in music score completion tasks based on musical input, which closely aligns with our model's objectives. Similarly, Music Transformer is capable of performing long-sequence symbolic music generation and music score completion tasks based on musical input. Additionally, we included the latest diffusion-based model, Cascaded-Diff, in our comparison. Cascaded-Diff is currently the only diffusion model capable of long-sequence symbolic music generation and demonstrates high music generation quality. We fine-tuned the baseline models on the MuseScore and POP909 datasets to ensure a fair comparison with our model. Furthermore, we conducted ablation studies to demonstrate the effectiveness of each component in our proposed Hi-ACG framework. In the experimental results, "GT" denotes ground truth, "MT" denotes Music Transformer, "BT" denotes BPE Transformer, and "CD" denotes Cascaded-Diff. In the ablation study section, "Full" denotes the complete framework, "SL" denotes the sketch loop, and "SP" denotes the semantic prediction.

Objective Evaluation To objectively evaluate of the music generated by various models, we design specialized music evaluation metrics. The evaluation metrics encompass four aspects, assessing both model-generated and real music from pitch, rhythm, harmony, and melody. Pitch evaluation employs information entropy to quantify note diversity within a musical piece. The information entropy is calculated as follows, where $p(i)$ represents the probability of note occurrence. Higher pitch entropy indicates greater pitch diversity with a more uniform distribution.

$$H_{\text{pitch}} = -\sum_{i=1}^{n} p(i) \log_2 p(i)$$

Similar to pitch evaluation, rhythm evaluation employs entropy to measure rhythmic complexity by analyzing the frequency and distribution of various note durations. Here, $p(j)$ denotes the probability of each duration type.

$$H_{\text{rhythm}} = -\sum_{j=1}^{n} p(j) \log_2 p(j)$$

	Pitch	Rhythm	Harmony	Melody	LLM score
GT	1.92	1.43	0.87	0.52	3.50
MT	**1.95**	**1.66**	0.94	0.41	2.25
BT	3.16	1.74	0.90	**0.55**	2.43
CD	3.26	2.36	0.91	0.66	**3.37**
w/o SL & SP	2.44	1.80	0.94	0.40	2.22
w/o SL	1.32	1.71	0.84	0.62	3.06
Full	1.43	1.69	**0.89**	0.60	3.10

Table 1: Objective evaluation results for 30-seconds unconditional music generation. Performance improves when Pitch, Rhythm, Harmony, and Melody values match the ground truth. Higher LLM scores show better performance.

	Pitch	Rhythm	Harmony	Melody	LLM score
GT	2.20	1.06	0.90	0.50	3.55
MT	3.49	3.19	0.92	0.29	2.29
BT	3.16	1.74	**0.90**	0.55	2.45
CD	3.38	2.30	0.91	0.90	2.89
w/o SL & SP	-	-	-	-	-
w/o SL	1.56	0.84	0.83	0.41	2.92
Full	**2.43**	**1.03**	**0.90**	**0.47**	**3.17**

Table 2: Objective evaluation results for 2-minutes unconditional music generation. Performance improves when Pitch, Rhythm, Harmony, and Melody values match the ground truth. Higher LLM scores show better performance.

Harmonic consistency evaluates the degree of matching between notes and tonality. We identify the musical tonality using Music21's (Cuthbert and Ariza 2010) key analysis algorithm, then calculate the proportion of notes that belong to the tonal distribution. The melodic smoothness metric is based on melodic fluency principles in music theory, assessing smoothness by analyzing the size of intervals between adjacent notes in the melody. Specifically, we define intervals exceeding a perfect fourth as large leaps and calculate the proportion of large leaps in the melody, where more frequent large leaps reduce the perceived musical quality. Additionally, we employ an LLM for music quality assessment. We use the Qwen3-235B-A22B model to evaluate musical quality, providing comprehensive scores considering multiple dimensions including melody, rhythm, and arrangement. The scoring employs the MOS (Mean Opinion Score) method with a score range of 1-5, where 1 is the lowest and 5 is the highest. We conduct objective evaluation across three tasks: 30-second short music generation, 2-minute long music generation, and conditional input long music generation. Short music generation evaluation represents local music quality, while long music generation assesses fluency, structure, and compositional completeness. Conditional music generation show the model's ability to continue music from given input, demonstrating music understanding and completion capabilities.

Subjective Evaluation In music generation tasks, subjective evaluation often provides a more intuitive reflection of the quality of the generated music's impact on listeners. We also conducted a comparison between our model and the baseline in subjective evaluation. The subjective evaluation employed the MOS method, where evaluators rate each music sample on a scale from 1 to 5, with higher scores indicating superior musical quality. Evaluators were blinded to which model generated each music sample. The evaluators were 79 volunteers with music education backgrounds , including 46 males and 33 females.

	Pitch	Rhythm	Harmony	Melody	LLM score
GT	2.20	1.06	0.90	0.50	3.55
MT	3.73	3.53	0.96	0.37	2.24
BT	3.89	3.20	0.88	0.66	2.33
CD	-	-	-	-	-
w/o SL & SP	-	-	-	-	-
w/o SL	2.69	1.90	0.99	0.33	3.05
Full	**2.19**	**1.27**	**0.91**	**0.43**	**3.30**

Table 3: Objective evaluation results for 2-minutes conditional music generation.Performance improves when Pitch, Rhythm, Harmony, and Melody values match the ground truth. Higher LLM scores show better performance.

	GT	MT	BT	CD	w/o SL & SP	w/o SL	Full
Score	3.31	1.96	2.05	2.91	1.85	2.52	**3.02**

Table 4: Subjective evaluation for 2-minutes music generation use MOS evaluation, higher score indicate superior performance.

Conclusion

Long-sequence symbolic music generation faces a critical challenge: error accumulation that degrades musical structure and fluency. To address this, we introduce the ACG paradigm with piano token representation and propose a hierarchical music generation framework. This approach enhances generation quality while enabling flexible conditional music generation. Our experiments demonstrate substantial improvements in both statistical metrics and overall music quality, with both objective metrics and subjective evaluations consistently validating our approach's superiority. The ACG paradigm represents a major breakthrough in long-sequence symbolic music generation, providing a flexible framework easily integrated into existing autoregressive models. The system adapts well to downstream tasks through fine-tuning and offers valuable insights for music understanding applications. These foundational principles pave the way for more sophisticated and controllable symbolic music generation systems.

While our method shows promising results, it currently lacks fine-grained control during generation, limiting the ability to dynamically adjust the autoregressive process for personalized and flexible music creation. Future work will address this limitation by integrating additional tokens that capture musical expression and structural elements into the piano token representation, which will substantially improve generation controllability. We also plan to extend the ACG paradigm to handle more complex scenarios, including multi-track symbolic music generation.

References

Beltagy, I.; Peters, M. E.; and Cohan, A. 2020. Longformer: The Long-Document Transformer. arXiv:2004.05150.

Briot, J.-P.; Hadjeres, G.; and Pachet, F.-D. 2017. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620*.

Child, R.; Gray, S.; Radford, A.; and Sutskever, I. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Conklin, D.; and Witten, I. H. 1995. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1): 51–73.

Cuthbert, M. S.; and Ariza, C. 2010. music21: A toolkit for computer-aided musicology and symbolic music analysis. In *11th International Society for Music Information Retrieval Conference (ISMIR 2010)*, 637–642.

Dong, H.-W.; Hsiao, W.-Y.; Yang, L.-C.; and Yang, Y.-H. 2017. MuseGAN: Multi-track Sequential Generative Adversarial Networks for Symbolic Music Generation and Accompaniment. arXiv:1709.06298.

Ebcioglu, K. 1988. An expert system for harmonizing four-part chorales. *Computer Music Journal*, 12(3): 43–51.

Eck, D.; and Schmidhuber, J. 2002. Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, 747–756. IEEE.

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

Fradet, N.; Gutowski, N.; Chhel, F.; and Briot, J.-P. 2023. Byte Pair Encoding for Symbolic Music. arXiv:2301.11975.

Goren, O.; Nachmani, E.; and Wolf, L. 2021. A-Muze-Net: Music Generation by Composing the Harmony based on the Generated Melody. arXiv:2111.12986.

Guo, Z.; Kang, J.; and Herremans, D. 2023. A Domain-Knowledge-Inspired Music Embedding Space and a Novel Attention Mechanism for Symbolic Music Modeling. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4): 5070–5077.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising Diffusion Probabilistic Models. arXiv:2006.11239.

Hsiao, W.-Y.; Liu, J.-Y.; Yeh, Y.-C.; and Yang, Y.-H. 2021a. Compound Word Transformer: Learning to Compose Full-Song Music over Dynamic Directed Hypergraphs. arXiv:2101.02402.

Hsiao, W.-Y.; Liu, J.-Y.; Yeh, Y.-C.; and Yang, Y.-H. 2021b. Compound Word Transformer: Learning to Compose Full-Song Music over Dynamic Directed Hypergraphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 64–72.

Huang, C.-Z. A.; Vaswani, A.; Uszkoreit, J.; Shazeer, N.; Simon, I.; Hawthorne, C.; Dai, A. M.; Hoffman, M. D.; Dinculescu, M.; and Eck, D. 2018. Music Transformer. arXiv:1809.04281.

Huang, Y.-S.; and Yang, Y.-H. 2020. Pop Music Transformer: Beat-based Modeling and Generation of Expressive Pop Piano Compositions. In *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*, 1180–1188. New York, NY, USA: Association for Computing Machinery.

Johnson, D. D. 2017. Generating polyphonic music using tied parallel networks. In *International Conference on Computational Intelligence in Music, Sound, Art and Design (EvoMUSART)*, 128–143. Springer.

Karystinaios, E.; and Widmer, G. 2024. GraphMuse: A Library for Symbolic Music Graph Processing. arXiv:2407.12671.

Li, Q.; Hu, Y.; Yao, F.; Xiao, C.; Liu, Z.; Sun, M.; and Shen, W. 2023. MUSER: A Multi-View Similar Case Retrieval Dataset. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, 5336–5340. ACM.

Liu, J.; Dong, Y.; Cheng, Z.; Zhang, X.; Li, X.; Yu, F.; and Sun, M. 2022. Symphony Generation with Permutation Invariant Language Model. arXiv:2205.05448.

Mittal, G.; Engel, J.; Hawthorne, C.; and Simon, I. 2021. Symbolic music generation with diffusion models. *arXiv preprint arXiv:2103.16091*.

Nistal, J.; Pasini, M.; Aouameur, C.; Grachten, M.; and Lattner, S. 2024. Diff-A-Riff: Musical Accompaniment Co-creation via Latent Diffusion Models. arXiv:2406.08384.

Oore, S.; Simon, I.; Dieleman, S.; Eck, D.; and Simonyan, K. 2020. This time with feeling: learning expressive musical performance. *Neural computing and applications*, 32(4): 955–967.

Plasser, M.; Peter, S.; and Widmer, G. 2023. Discrete Diffusion Probabilistic Models for Symbolic Music Generation. arXiv:2305.09489.

Qu, X.; Bai, Y.; Ma, Y.; Zhou, Z.; Lo, K. M.; Liu, J.; Yuan, R.; Min, L.; Liu, X.; Zhang, T.; Du, X.; Guo, S.; Liang, Y.; Li, Y.; Wu, S.; Zhou, J.; Zheng, T.; Ma, Z.; Han, F.; Xue, W.; Xia, G.; Benetos, E.; Yue, X.; Lin, C.; Tan, X.; Huang, S. W.; Fu, J.; and Zhang, G. 2024. MuPT: A Generative Symbolic Music Pretrained Transformer. arXiv:2404.06393.

Roberts, A.; Engel, J.; Raffel, C.; Hawthorne, C.; and Eck, D. 2018. A hierarchical latent vector model for learning long-term structure in music. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 4364–4373. PMLR.

Sturm, B. L.; Santos, J. F.; Ben-Tal, O.; and Korshunova, I. 2016. Music transcription modelling and composition using deep learning. *arXiv preprint arXiv:1604.08723*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, volume 30. Curran Associates, Inc.

Wang, Z.; Min, L.; and Xia, G. 2024. Whole-Song Hierarchical Generation of Symbolic Music Using Cascaded Diffusion Models. arXiv:2405.09901.

Wu, S.; Li, X.; Yu, F.; and Sun, M. 2023. TunesFormer: Forming Irish Tunes with Control Codes by Bar Patching. arXiv:2301.02884.

Yu, B.; Lu, P.; Wang, R.; Hu, W.; Tan, X.; Ye, W.; Zhang, S.; Qin, T.; and Liu, T.-Y. 2022. Museformer: Transformer with Fine- and Coarse-Grained Attention for Music Generation. arXiv:2210.10349.

Yu, L.; Zhang, W.; Wang, J.; and Yu, Y. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. arXiv:1609.05473.

Yuan, R.; Lin, H.; Wang, Y.; Tian, Z.; Wu, S.; Shen, T.; Zhang, G.; Wu, Y.; Liu, C.; Zhou, Z.; Ma, Z.; Xue, L.; Wang, Z.; Liu, Q.; Zheng, T.; Li, Y.; Ma, Y.; Liang, Y.; Chi, X.; Liu, R.; Wang, Z.; Li, P.; Wu, J.; Lin, C.; Liu, Q.; Jiang, T.; Huang, W.; Chen, W.; Benetos, E.; Fu, J.; Xia, G.; Dannenberg, R.; Xue, W.; Kang, S.; and Guo, Y. 2024. ChatMusician: Understanding and Generating Music Intrinsically with LLM. arXiv:2402.16153.

Zhao, B.; Li, X.; and Lu, X. 2019. Hierarchical Recurrent Neural Network for Video Summarization. arXiv:1904.12251.

Reproducibility Checklist

1. General Paper Structure

1.1. Includes a conceptual outline and/or pseudocode description of AI methods introduced (yes/partial/no/NA) yes

1.2. Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results (yes/no) yes

1.3. Provides well-marked pedagogical references for less-familiar readers to gain background necessary to replicate the paper (yes/no) yes

2. Theoretical Contributions

2.1. Does this paper make theoretical contributions? (yes/no) yes

If yes, please address the following points:

2.2. All assumptions and restrictions are stated clearly and formally (yes/partial/no) yes

2.3. All novel claims are stated formally (e.g., in theorem statements) (yes/partial/no) yes

2.4. Proofs of all novel claims are included (yes/partial/no) yes

2.5. Proof sketches or intuitions are given for complex and/or novel results (yes/partial/no) yes

2.6. Appropriate citations to theoretical tools used are given (yes/partial/no) yes

2.7. All theoretical claims are demonstrated empirically to hold (yes/partial/no/NA) yes

2.8. All experimental code used to eliminate or disprove claims is included (yes/no/NA) yes

3. Dataset Usage

3.1. Does this paper rely on one or more datasets? (yes/no) yes

If yes, please address the following points:

3.2. A motivation is given for why the experiments are conducted on the selected datasets (yes/partial/no/NA) yes

3.3. All novel datasets introduced in this paper are included in a data appendix (yes/partial/no/NA) yes

3.4. All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no/NA) yes

3.5. All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations (yes/no/NA) yes

3.6. All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available (yes/partial/no/NA) yes

3.7. All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing (yes/partial/no/NA) yes

4. Computational Experiments

4.1. Does this paper include computational experiments? (yes/no) yes

If yes, please address the following points:

4.2. This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting (yes/partial/no/NA) yes

4.3. Any code required for pre-processing data is included in the appendix (yes/partial/no) yes

4.4. All source code required for conducting and analyzing the experiments is included in a code appendix (yes/partial/no) yes

4.5. All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes (yes/partial/no) yes

4.6. All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from (yes/partial/no) yes

4.7. If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results (yes/partial/no/NA) yes

4.8. This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks (yes/partial/no) yes

4.9. This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics (yes/partial/no) yes

4.10. This paper states the number of algorithm runs used to compute each reported result (yes/no) yes

4.11. Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information (yes/no) yes

4.12. The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank) (yes/partial/no) yes

4.13. This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments (yes/partial/no/NA) yes