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

Boyu Cao[1,*] **Lekai Qian**[1,*] **Dehan Li**[1] **Haoyu Gu**[1] **Mingda Xu**[1] **Qi Liu**[1,†]

[1]South China University of Technology, School of Future Technology

[*]Equal contribution. [†]Correspondence: drliuqi@scut.edu.cn

Abstract

Generating long sequences with structural coherence remains a fundamental challenge for autoregressive models across sequential generation tasks. In symbolic music generation, this challenge is particularly pronounced, as existing methods are constrained by the severe error accumulation inherent in autoregressive models, leading to poor performance in music quality and structural integrity. In this paper, we propose the Anchored Cyclic Generation (ACG) paradigm, which relies on anchor features from previously generated musical content 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 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.

1 Introduction

Autoregressive sequence modeling has achieved remarkable success across various domains, from natural language processing to structured content generation. However, maintaining long-term coherence and structural integrity in extended sequences remains a persistent challenge due to error accumulation during iterative generation. Symbolic music generation exemplifies this challenge: as a core branch of music generation, it produces discrete musical representations with structured, interpretable characteristics (Briot et al., 2017), yet modeling long-sequence symbolic music has become a primary challenge with the rapid development of deep learning technologies.

Long-sequence symbolic music generation requires maintaining both local coherence and global structural integrity. Autoregressive models, the mainstream approach, predict subsequent segments based on historical content but face significant limitations. Early RNN/LSTM approaches exhibit degraded quality and stylistic drift in longer sequences, making it difficult to maintain long-term consistency. Attention-based methods like Transformers, combined with MIDI event encoding (Oore et al., 2020) or ABC notation, perform well on short sequences but face exponentially growing computational complexity (Child et al., 2019) and severe error accumulation as length increases. Diffusion models (Mittal et al., 2021) offer an alternative but struggle to generate complete long-sequence music efficiently.

Addressing these challenges, we propose the Anchored Cyclic Generation (ACG) paradigm, which introduces determined musical content as anchors in each generation cycle to recalibrate the generation process, effectively mitigating error accumulation and achieving smooth transitions between musical segments. ACG ensures local musical quality while maintaining structural integrity of long-sequence music, with significant advantages in time complexity. Based on ACG, we further propose a Hierarchical Anchored Cyclic Generation (Hi-ACG) framework with a novel piano token representation. The framework adopts a hierarchical

Findings of the Association for Computational Linguistics: ACL 2026, pages 31463–31476

strategy: a sketch loop captures high-level semantic features such as modality, harmonic progression, and overall structure, providing global guidance for subsequent processes; a refinement loop then generates detailed note-level content to ensure local coherence and expressiveness. Experiments demonstrate that Hi-ACG maintains long-term structural and stylistic consistency while achieving precise duration control.

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 empirical analysis and experimental validation, our proposed ACG paradigm and Hi-ACG framework 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.

2 Related Work

2.1 Symbolic Music Generation

Symbolic music generation aims to automatically generate discrete musical representations with musicality and interpretability. Early approaches based on rules (Ebcioglu, 1988) and statistical modeling (Conklin and Witten, 1995) laid the foundation for data-driven methods. With deep learning, RNNs (Eck and Schmidhuber, 2002; Sturm et al., 2016), VAE-based models like MusicVAE (Roberts et al., 2018), and adversarial approaches like MuseGAN (Dong et al., 2017) became mainstream, though they struggled with long sequences.

Recently, Transformers (Vaswani et al., 2017) and diffusion models (Ho et al., 2020) have advanced the field. RIPO Transformer (Guo et al., 2023) enhanced melody modeling with novel attention mechanisms. TunesFormer (Wu et al., 2023) enabled bar-level controlled generation. ChatMusician (Yuan et al., 2024) demonstrated how language models can understand and generate music through unified text-music pretraining.

2.2 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.

The challenge of modeling long sequences is shared across domains. In natural language processing, methods such as Longformer (Beltagy et al., 2020) and hierarchical text generation approaches have addressed similar issues of computational complexity and long-range coherence. Our work draws inspiration from these cross-domain advances while addressing the unique structural requirements of symbolic music.

2.2.1 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 et al., 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.

2.2.2 Diffusion-based Methods

Diffusion models offer a non-autoregressive generation approach with strong fidelity. Discrete diffusion models (Plasser et al., 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 et al., 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 for long sequences due to multi-step sampling, and often struggle with maintaining coherence and duration control.

2.3 Hierarchical Music Generation

Hierarchical music generation addresses long-sequence challenges by decomposing generation into multiple levels (e.g., sections, phrases, notes). Early models like Hierarchical RNN (Zhao et al., 2019) used multi-layer recurrent structures. SymphonyNet (Liu et al., 2022) modeled movements, phrases, and notes for symphonic music. Cascaded-Diff (Wang et al., 2024) integrated structural language modeling with diffusion techniques. Despite progress, current methods still face issues in inter-level information flow and precise duration control.

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.

3 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.

3.1 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.

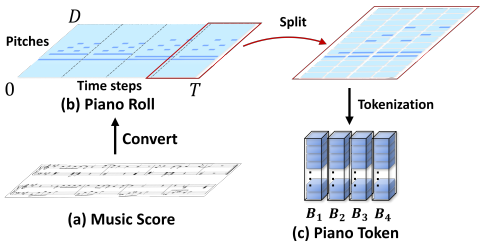

Figure 1: Converting musical scores to piano tokens. (a) Example musical score. (b) Piano roll with T time steps and D pitch dimensions. The red-boxed section demonstrates split details. (c) Piano token representation, where each sub-unit of the split piano roll is tokenized into a piano token. Piano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

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 representation 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.

3.2 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 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 is shown in Figure 3. Compared to traditional

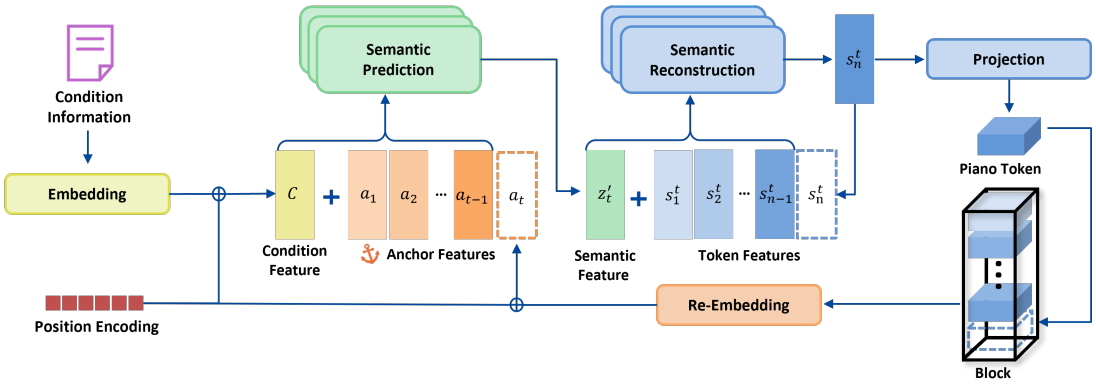

Figure 2: ACG Paradigm Architecture. The embedding layer encodes the conditional information into feature vectors, concatenated with anchor features from existing content (anchor features are absent at the initial time step) and feeds into the semantic prediction model. The semantic prediction model 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.

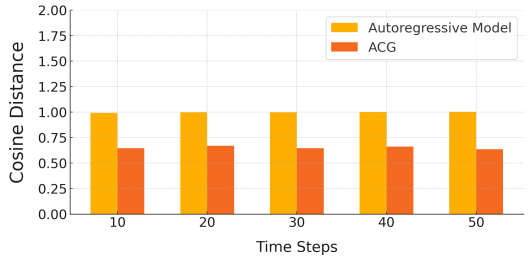

Figure 3: Cosine distances between predicted and ground truth feature vectors for the ACG paradigm and conventional autoregressive models across iterations. The ACG paradigm consistently achieves lower cosine distances compared to conventional autoregressive models.

autoregressive models, the ACG paradigm achieves an average reduction of 34.7% in cosine distance between predicted feature vectors and ground-truth semantic vectors. Appendix B provides an intuition for why anchoring confirmed content can stabilize subsequent predictions.

Together, these results provide empirical evidence that the proposed ACG paradigm reduces feature drift relative to conventional autoregressive generation, thereby improving generation stability. 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.

3.3 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

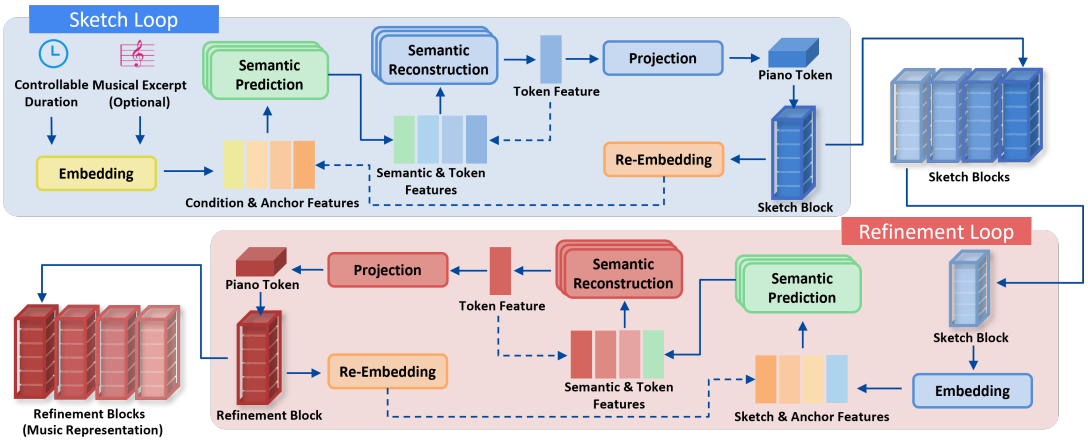

Figure 4: The Hi-ACG framework, which comprises a sketch loop and a refinement loop. The sketch loop takes duration condition and optional musical constraints as input, generating a musical sketch with coarse-grained information for the entire composition using block-by-block processing. The refinement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

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.

4 Experiment

To validate the Hi-ACG framework's effectiveness, we conduct comprehensive objective and subjective experiments evaluating the generation quality.

This section presents the experimental setup and evaluation results.

4.1 Experimental Setup

4.1.1 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 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.

4.1.2 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.

4.2 Evaluation

4.2.1 Baseline

We compare against Transformer-based models (Music Transformer, BPE Transformer) and diffusion-based model (Cascaded-Diff), all fine-tuned on the same datasets for fair comparison. We also conduct ablation studies on Hi-ACG components. In tables, “GT" denotes ground truth, “MT" denotes Music Transformer, “BT" denotes BPE Transformer, “CD" denotes Cascaded-Diff, “Full" denotes the complete framework, “SL" denotes the sketch loop, and “SP" denotes the semantic prediction.

4.2.2 Objective Evaluation

To objectively evaluate 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.

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

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. We conduct objective evaluation across three tasks: 30-second short music generation, 2-minute long music generation, and conditional long music generation. Short music generation primarily reflects local musical quality, while long music generation assesses fluency, structure, and compositional completeness. Conditional music generation shows the model’s ability to continue music from given input, demonstrating music understanding and completion capabilities. Because LLM-based evaluation is not yet a widely accepted primary protocol for symbolic music generation, we treat it as supplementary evidence rather than a main claim and report those results in Appendix E.

4.2.3 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

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

Table 1: Objective evaluation results for 30-second unconditional music generation. Performance improves when the Pitch, Rhythm, Harmony, and Melody values are closer to the ground truth.

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

Table 2: Objective evaluation results for 2-minute unconditional music generation. Performance improves when the Pitch, Rhythm, Harmony, and Melody values are closer to the ground truth.

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.

Table 4 reports the subjective evaluation results on long-form music generation. Among all generated systems, the full Hi-ACG model achieves the best performance across all three dimensions, obtaining an overall score of 3.02, a smoothness score of 3.10, and a richness score of 3.11. It consistently outperforms all baseline methods, indicating that our framework produces music that is not only preferred in overall quality, but also perceived as more coherent and musically expressive by human listeners. Although a gap to ground-truth music still remains, the full model is substantially closer to GT than the competing systems, especially in smoothness and richness, suggesting that Hi-ACG is more effective at maintaining long-range continuity while generating diverse musical content.

Notably, CD is the strongest baseline in subjective evaluation, but it still falls short of the full Hi-ACG model on all three criteria. This indicates that the advantage of our method is not limited to a single perceptual aspect; instead, it improves both structural continuity and musical richness in a consistent manner. The relatively large gains in smoothness further suggest that the proposed framework is particularly effective in reducing abrupt transitions and preserving coherence over long musical horizons. The ablation results show a similar trend: removing both SL and SP leads to the largest performance drop, while removing only SL partially recovers the scores but still remains clearly below the full model. These findings verify that both components contribute to human-perceived quality, and that their combination is important for producing smooth and rich long-form musical structures.

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

Table 3: Objective evaluation results for 2-minute conditional music generation. Performance improves when the Pitch, Rhythm, Harmony, and Melody values are closer to the ground truth.

Score	GT	MT	BT	CD	w/o SL & SP	w/o SL	Full
Overall	3.31	1.96	2.05	2.91	1.85	2.52	**3.02**
Smoothness	3.58	1.77	1.93	2.97	1.82	2.58	**3.10**
Richness	3.52	1.83	2.06	2.94	1.76	2.50	**3.11**

Table 4: Subjective evaluation results for long-form music generation. We report mean human ratings on overall quality, smoothness, and richness for ground-truth music (GT), baseline systems, and ablated variants of our method. The full model consistently receives the highest scores among all generated systems.

5 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 promising framework in long-sequence symbolic music generation, pro-

viding a flexible framework easily integrated into existing autoregressive models. The system shows potential for 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 demonstrated on symbolic music, the core principle of ACG—using confirmed content as anchors to mitigate error accumulation—is broadly applicable to other long-sequence generation tasks, including structured text generation and hierarchical content synthesis, opening promising directions for future research.

Limitations

Our method currently lacks fine-grained control during generation, limiting dynamic adjustment for personalized music creation. Future work will integrate additional tokens capturing musical expression and structural elements to improve controllability. Additionally, while the piano token representation achieves efficient compression, it may lose subtle timing nuances present in event-based representations. The current framework focuses on piano music; extending to diverse instruments and timbres requires further investigation. We also plan to extend ACG to multi-track symbolic music generation and explore its application to other sequential generation domains.

Acknowledgments

We thank the reviewers and area chairs for their constructive feedback. We used large language models only for linguistic and presentational assistance, including terminology checking, grammar correction, and improving sentence flow while preserving the original scientific meaning. They were not used for research ideation, experimental design, data analysis, or scientific content generation. All scientific insights, methodological decisions, experiments, and conclusions in this paper are the authors' own.

References

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Preprint*, arXiv:2004.05150.

Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. 2017. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

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

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

Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. 2017. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. *Preprint*, arXiv:1709.06298.

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

Douglas Eck and Jürgen Schmidhuber. 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*, pages 747–756. IEEE.

Nathan Fradet, Nicolas Gutowski, Fabien Chhel, and Jean-Pierre Briot. 2023. Byte pair encoding for symbolic music. *Preprint*, arXiv:2301.11975.

Zixun Guo, Jaeyong Kang, and Dorien Herremans. 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.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Preprint*, arXiv:2006.11239.

Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. 2021a. Compound word transformer: Learning to compose full-song music over dynamic directed hypergraphs. *Preprint*, arXiv:2101.02402.

Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. 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, pages 64–72.

Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. 2018. Music transformer. *Preprint*, arXiv:1809.04281.

Yu-Siang Huang and Yi-Hsuan Yang. 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)*, pages 1180–1188, New York, NY, USA. Association for Computing Machinery.

Jiafeng Liu, Yuanliang Dong, Zehua Cheng, Xinran Zhang, Xiaobing Li, Feng Yu, and Maosong Sun. 2022. Symphony generation with permutation invariant language model. *Preprint*, arXiv:2205.05448.

Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. 2021. Symbolic music generation with diffusion models. *arXiv preprint arXiv:2103.16091*.

Javier Nistal, Marco Pasini, Cyran Aouameur, Maarten Grachten, and Stefan Lattner. 2024. Diff-a-riff: Musical accompaniment co-creation via latent diffusion models. *Preprint*, arXiv:2406.08384.

Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. 2020. This time with feeling: learning expressive musical performance. *Neural computing and applications*, 32(4):955–967.

Matthias Plasser, Silvan Peter, and Gerhard Widmer. 2023. Discrete diffusion probabilistic models for symbolic music generation. *Preprint*, arXiv:2305.09489.

Xingwei Qu, Yuelin Bai, Yinghao Ma, Ziya Zhou, Ka Man Lo, Jiaheng Liu, Ruibin Yuan, Lejun Min, Xueling Liu, Tianyu Zhang, Xinrun Du, Shuyue Guo, Yiming Liang, Yizhi Li, Shangda Wu, Junting Zhou, Tianyu Zheng, Ziyang Ma, Fengze Han, and 9 others. 2024. Mupt: A generative symbolic music pretrained transformer. *Preprint*, arXiv:2404.06393.

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

Bob L. Sturm, João Felipe Santos, Oded Ben-Tal, and Iryna Korshunova. 2016. Music transcription modelling and composition using deep learning. *arXiv preprint arXiv:1604.08723*.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, volume 30. Curran Associates, Inc.

Ziyu Wang, Lejun Min, and Gus Xia. 2024. Whole-song hierarchical generation of symbolic music using cascaded diffusion models. *Preprint*, arXiv:2405.09901.

Shangda Wu, Xiaobing Li, Feng Yu, and Maosong Sun. 2023. Tunesformer: Forming irish tunes with control codes by bar patching. *Preprint*, arXiv:2301.02884.

Botao Yu, Peiling Lu, Rui Wang, Wei Hu, Xu Tan, Wei Ye, Shikun Zhang, Tao Qin, and Tie-Yan Liu. 2022. Museformer: Transformer with fine- and coarse-grained attention for music generation. *Preprint*, arXiv:2210.10349.

Ruibin Yuan, Hanfeng Lin, Yi Wang, Zeyue Tian, Shangda Wu, Tianhao Shen, Ge Zhang, Yuhang Wu, Cong Liu, Ziya Zhou, Ziyang Ma, Liumeng Xue, Ziyu Wang, Qin Liu, Tianyu Zheng, Yizhi Li, Yinghao Ma, Yiming Liang, Xiaowei Chi, and 16 others. 2024. Chatmusician: Understanding and generating music intrinsically with llm. *Preprint*, arXiv:2402.16153.

Bin Zhao, Xuelong Li, and Xiaoqiang Lu. 2019. Hierarchical recurrent neural network for video summarization. *Preprint*, arXiv:1904.12251.

Algorithm 1 Piano roll to piano tokens conversion

Require: Piano roll matrix $\mathbf{P} \in \{0,1\}^{88\times W}$
Ensure: Token matrix $\mathbf{S} \in \{0,1,\ldots,255\}^{W/4\times 44}$
$h_p \leftarrow 2, w_p \leftarrow 4$ ▷ Patch dimensions
$N_r \leftarrow 88/h_p = 44$, $N_c \leftarrow W/w_p$
Initialize $\mathbf{S} \in \mathbb{Z}^{N_c\times N_r}$
for $j = 0$ **to** $N_c - 1$ **do**
  for $i = 0$ **to** $N_r - 1$ **do**
    $\mathbf{patch} \leftarrow \mathbf{P}[i{\cdot}h_p : (i{+}1){\cdot}h_p, j{\cdot}w_p : (j{+}1){\cdot}w_p]$
    $\mathbf{flat} \leftarrow \text{flatten}(\mathbf{patch})$
    $token \leftarrow 0$
    for $k = 0$ **to** 7 **do**
      $token \leftarrow (token \ll 1) \vee \mathbf{flat}[k]$
    end for
    $\mathbf{S}[j,i] \leftarrow token$
  end for
end for
return S

Algorithm 2 Piano tokens to piano roll conversion

Require: Token matrix $\mathbf{S} \in \{0,1,\ldots,255\}^{N_c\times N_r}$
Ensure: Reconstructed piano roll $\mathbf{P}' \in \{0,1\}^{88\times 4N_c}$
$h_p \leftarrow 2, w_p \leftarrow 4$
Initialize $\mathbf{P}' \in \{0,1\}^{N_r\cdot h_p \times N_c\cdot w_p}$
for $j = 0$ **to** $N_c - 1$ **do**
  for $i = 0$ **to** $N_r - 1$ **do**
    $token \leftarrow \mathbf{S}[j,i]$
    $\mathbf{binary} \leftarrow \text{int2bin}(token, 8)$
    $\mathbf{patch} \leftarrow \text{reshape}(\mathbf{binary}, (h_p, w_p))$
    $\mathbf{P}'[i\cdot h_p : (i+1)\cdot h_p, j\cdot w_p : (j+1)\cdot w_p] \leftarrow \mathbf{patch}$
  end for
end for
return $\mathbf{P}'$

A Piano Token Conversion Algorithm

This section provides the conversion algorithms for the piano token representation. Algorithm 1 describes how to convert a piano roll representation into piano tokens, and Algorithm 2 shows the reverse conversion.

B Intuition on Why Anchoring Reduces Error Drift

We provide an intuition for why ACG can mitigate error accumulation, rather than a formal proof that one loss is always lower than another.

Let B_t^* denote the ideal discrete block at step t, and let $a_t^* = f(B_t^*)$ be its corresponding ideal anchor feature under the re-embedding map f. In ACG, the actual anchor a_t is obtained by re-embedding a confirmed discrete block B_t. Because B_t is discrete and fixed once generated, the mismatch between B_t and B_t^* is naturally limited by the quantization resolution of the tokenization. Let $\delta_{\max}$ denote this maximum block-level quantization mismatch. If the re-embedding map f is locally L_f-Lipschitz, then the induced anchor mismatch satisfies

$$\begin{aligned}\|a_t - a_t^*\| &= \|f(B_t) - f(B_t^*)\| \\ &\le L_f \|B_t - B_t^*\| \le L_f\, \delta_{\max}.\end{aligned}$$

Let ε_t' denote the step-local prediction error under anchoring, and let $\varepsilon_{\text{step}}$ denote the residual error introduced by the semantic prediction and reconstruction modules at the current step. Then the total step-local mismatch under ACG can be bounded as

$$\|\varepsilon_t'\| \le L_f\, \delta_{\max} + \varepsilon_{\text{step}}.$$

This bound is useful because it depends on the local quantization mismatch and the current-step model error, rather than on an unbounded accumulation of previously generated continuous states. In other words, re-embedding confirmed discrete content limits how much historical deviation can be propagated into the next prediction.

By contrast, conventional autoregressive decoding conditions directly on previously generated states, so deviations in earlier steps can continue to perturb subsequent predictions. The anchoring mechanism does not eliminate error, but it can restrict step-to-step drift by repeatedly projecting the generation history back to a confirmed discrete representation before producing the next anchor feature.

Therefore, the role of Appendix B is to provide a bounded-error intuition for ACG rather than a theorem claiming that $L_2 \le L_1$ for all settings. The quantitative evidence for reduced drift is given empirically in Figure 3 of the main text and in Figures 5 and 6 of Appendix D, where ACG consistently yields lower cosine distance than conventional autoregressive decoding.

C Time Complexity Analysis

Conventional autoregressive models have time complexity $\mathcal{O}(L^2 \cdot d)$ where L is sequence length and d is hidden dimension.

The ACG paradigm decomposes this into two stages:

- **Semantic Prediction**: $\mathcal{C}_{\text{sem}} = \mathcal{O}(L_{sem}^2 \cdot d)$
- **Semantic Reconstruction**: $\mathcal{C}_{\text{rec}} = L_{sem} \times \mathcal{O}(L_{rec}^2 \cdot d)$

Total ACG complexity:

$$\mathcal{C}_{\text{ACG}} = \mathcal{O}(L_{sem}^2 \cdot d) + L_{sem} \times \mathcal{O}(L_{rec}^2 \cdot d)$$

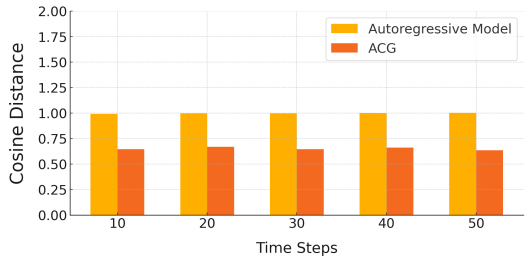

Figure 5: Cosine distance comparison across 50 iterative steps.

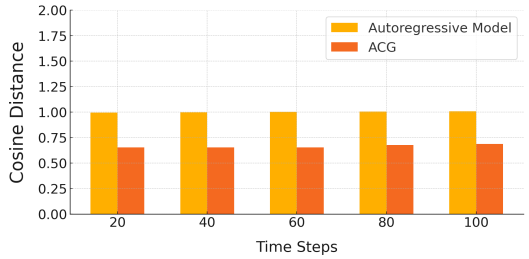

Figure 6: Cosine distance comparison across 100 iterative steps.

Given $L = L_{sem} \times L_{rec}$ and fixed L_{rec}, the speedup factor is $\mathcal{O}(L_{rec}^2)$, making ACG particularly effective for long sequences.

D Extended Cosine Distance Comparison

Figure 5 and Figure 6 show cosine distances between predicted and ground truth features for 50 and 100 timesteps respectively, demonstrating ACG's consistent advantage.

E Supplementary LLM-based Reference Evaluation

We additionally report LLM-based evaluation as supplementary evidence. Since LLM-as-a-judge is not yet a universally accepted primary evaluation protocol for symbolic music generation, we do not use these scores as central evidence in the main text. Specifically, we use Qwen3-235B-A22B (Team, 2025) to rate generated music with a MOS-style score from 1 to 5 by considering melody, rhythm, and arrangement. These results should be interpreted as a reference complementing the automatic metrics and the human listening test reported in the main paper.

F Ablation Studies

F.1 Architectural Ablation

We investigate the effects of model size with Small (4+8 layers), Middle (5+10 layers), and Large (6+12 layers) configurations. Results in Tables 6, 7, and 8 show the Large model achieves best performance.

	30-second un-con	2-minute un-con	2-minute con
GT	3.50	3.55	3.55
MT	2.25	2.29	2.24
BT	2.43	2.45	2.33
CD	3.37	2.89	-
w/o SL & SP	2.22	-	-
w/o SL	3.06	2.92	3.05
Full	**3.10**	**3.17**	**3.30**

Table 5: Supplementary LLM-based reference evaluation using Qwen3-235B-A22B. Higher scores indicate better quality.

	Pitch	Rhythm	Harmony	Melody	LLM
GT	1.92	1.43	0.87	0.52	3.50
Small	1.40	1.75	0.77	0.40	3.01
Middle	1.41	**1.68**	0.84	**0.59**	3.08
Large	**1.43**	1.69	**0.89**	0.60	**3.10**

Table 6: Architectural ablation on 30-second generation.

F.2 Hyperparameter Ablation

We investigate patch dimensions (d, t) affecting vocabulary size and sequence lengths. Tables 9, 10, and 11 show $\text{Patch}_{2,4}$ achieves optimal balance.

G Generation Examples

Figures 7 to 9 show piano roll visualizations of music generated by Hi-ACG.

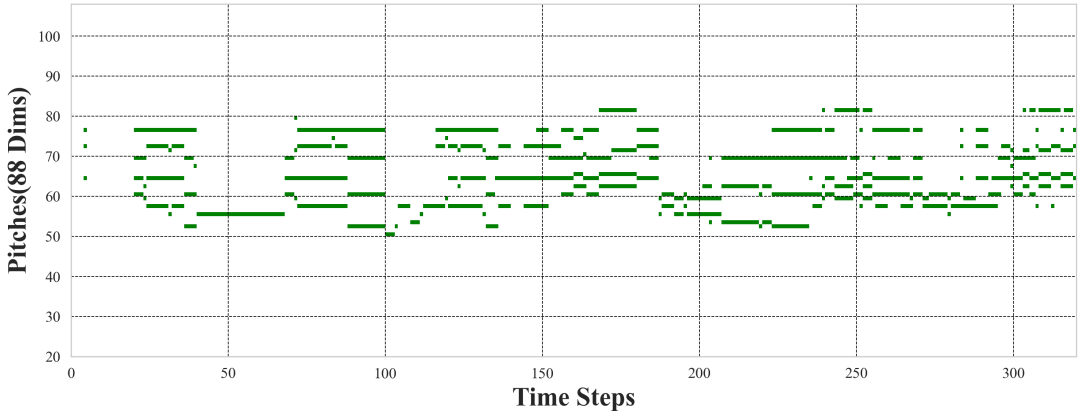

Figure 7: Example of 30-second music generated by Hi-ACG.

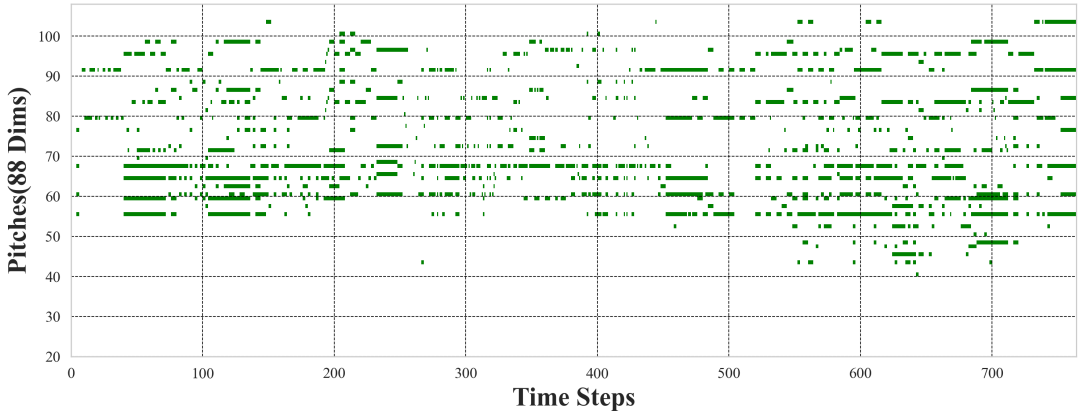

Figure 8: Example 1 of 2-minute unconditional music generated by Hi-ACG.

	Pitch	Rhythm	Harmony	Melody	LLM
GT	2.20	1.06	0.90	0.50	3.55
Small	2.45	**1.08**	0.94	0.55	3.11
Middle	2.51	1.03	0.92	0.46	3.12
Large	**2.43**	1.03	**0.90**	**0.47**	**3.17**

Table 7: Architectural ablation on 2-minute unconditional generation.

	Pitch	Rhythm	Harmony	Melody	LLM
GT	2.20	1.06	0.90	0.50	3.55
Small	2.16	1.33	0.82	0.40	3.25
Middle	2.15	1.35	0.87	0.42	3.28
Large	**2.19**	**1.27**	**0.91**	**0.43**	**3.30**

Table 8: Architectural ablation on 2-minute conditional generation.

	Pitch	Rhythm	Harmony	Melody	LLM
GT	1.92	1.43	0.87	0.52	3.50
$\text{Patch}_{1,4}$	2.45	1.88	0.77	0.42	2.89
$\text{Patch}_{2,4}$	**1.43**	1.69	**0.89**	0.60	**3.10**
$\text{Patch}_{3,4}$	2.53	**1.51**	0.82	**0.55**	3.06

Table 9: Hyperparameter ablation on 30-second generation.

	Pitch	Rhythm	Harmony	Melody	LLM
GT	2.20	1.06	0.90	0.50	3.55
$\text{Patch}_{1,4}$	2.81	1.22	0.95	**0.53**	3.02
$\text{Patch}_{2,4}$	**2.43**	1.03	**0.90**	0.47	**3.17**
$\text{Patch}_{3,4}$	2.45	**1.02**	0.91	0.45	3.14

Table 10: Hyperparameter ablation on 2-minute unconditional generation.

	Pitch	Rhythm	Harmony	Melody	LLM
GT	2.20	1.06	0.90	0.50	3.55
$\text{Patch}_{1,4}$	2.24	1.34	0.95	0.56	3.19
$\text{Patch}_{2,4}$	**2.19**	**1.27**	0.91	0.43	**3.30**
$\text{Patch}_{3,4}$	2.18	1.20	**0.90**	**0.49**	3.22

Table 11: Hyperparameter ablation on 2-minute conditional generation.

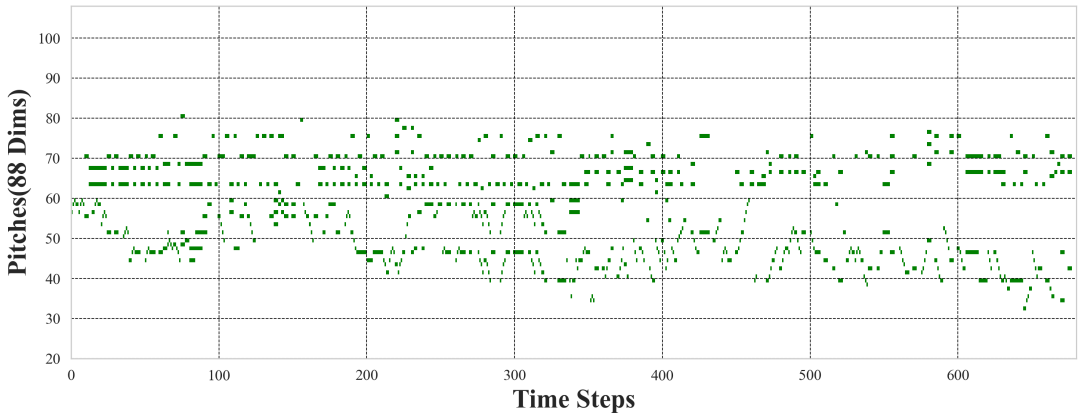

Figure 9: Example of 2-minute conditional music generated by Hi-ACG.