
Unifying Symbolic Music Arrangement: Track-Aware Reconstruction and Structured Tokenization

Longshen Ou[#] Jingwei Zhao[#] Ziyu Wang^{b,‡} Gus Xia[‡]
Qihao Liang[#] Torin Hopkins[#] Ye Wang[#]

[#]Sound and Music Computing Lab, School of Computing, NUS

^bCourant Institute of Mathematical Sciences, New York University

[‡]Music X Lab, MBZUAI

Abstract

We present a unified framework for automatic multitrack music arrangement that enables a single pre-trained symbolic music model to handle diverse arrangement scenarios, including reinterpretation, simplification, and additive generation. At its core is a segment-level reconstruction objective operating on token-level disentangled content and style, allowing for flexible any-to-any instrumentation transformations at inference time. To support track-wise modeling, we introduce *REMI-z*, a structured tokenization scheme for multitrack symbolic music that enhances modeling efficiency and effectiveness for both arrangement tasks and unconditional generation. Our method outperforms task-specific state-of-the-art models on representative tasks in different arrangement scenarios—band arrangement, piano reduction, and drum arrangement, in both objective metrics and perceptual evaluations. Taken together, our framework demonstrates strong generality and suggests broader applicability in symbolic music-to-music transformation.¹

1 Introduction

Music arrangement is the art of adapting compositions for performance contexts that differ from their original forms [4]. It plays a central role in many music creation process, including professional production, live performance, music education, and digital content creation. Automating this process can expand music accessibility and accelerate music creation. Although arrangement forms vary—e.g., rewriting for different instruments (reinterpretation) [9], simplifying for solo performance (reduction) [29, 30], or adding new tracks (additive generation) [20, 19]—they share a common structure: generating new music tracks conditioned on existing ones under explicit content and instrument constraints. However, prior work typically addresses each task independently [42, 30, 29, 19], using specialized model architectures and training schemes. Such designs lack cross-task generality, increase implementation cost, and fail to leverage the musical knowledge learned by large-scale pre-trained symbolic models that could potentially improve the arrangement quality.

Meanwhile, generative modeling of symbolic music, i.e., music in notation-based formats, has advanced rapidly with large-scale pre-trained models that capture rich musical styles and structures via autoregressive modeling [17, 8, 23, 37, 32, 22]. Inspired by natural language processing, these models scale to billions of parameters and are trained on vast corpora. Although their unconditional generation quality has improved substantially, their applications in real-world conditional generation remains relatively limited. Prior work primarily targets coarse-level control, such as style [3, 17], structure [40], polyphony level [22], or sentiment [18], while fine-grained conditioning on existing musical content—precisely what arrangement tasks demand—remains underexplored.

¹Demos and code: https://www.oulongshen.xyz/automatic_arrangement.



Figure 1: Overview of the proposed unified framework for each arrangement task. The symbol $\oplus$ denotes concatenation of component sequences. Music segments are decomposed into three subsequences: *instruments*, *content*, and *target-side history*. These components form the *condition sequence*, with the relevant tracks from the original music as the *target sequence*. The model is trained to reconstruct the music from these components.

This underexploration is likely not due to the absence of pre-trained models or their representational power, but to the lack of training objectives that support flexible adaptation to the diverse requirements of arrangement tasks. While sequence-to-sequence fine-tuning is conceptually a natural solution, it relies on parallel datasets—collections of the same piece arranged for different instrumentations (e.g., orchestra and piano)—which are extremely scarce. Even when adopting some level of internal parallel data [29, 30], these datasets constrain output directions to fixed mappings (e.g., orchestra to piano). In contrast, real-world arrangement demands far greater flexibility: the ability to transform arbitrary input instrumentations into arbitrary targets—i.e., support any-to-any transformations.

To address these limitations, we propose a unified framework for symbolic music arrangement that enables a pre-trained symbolic generative model to be fine-tuned across diverse tasks through a single self-supervised training pipeline (Figure 1). This facilitates the transfer of musical knowledge from generative pre-training to improve arrangement quality, while reducing the need for task-specific model design. To achieve this, we designed a context-aware, segment-level reconstruction objective: the model reconstructs multitrack music from its disentangled components, including content (notes executed) and style (instrumentation). We further observe that strictly time-ordered tokenizations (e.g., REMI+ [31]) introduce redundancy and fragment track content (detailed in §2.2), hindering instrument-level control and modeling. To address this, we propose a structured tokenization scheme that relaxes global time ordering while promoting track-wise continuity, enabling more consistent encoding across musical contexts. Our main contributions are as follows:

- We propose a **unified framework for symbolic music arrangement** that supports flexible instrumentation transformation across multiple typical arrangement scenarios, all without requiring parallel data. Central to our approach is a shared reconstruction objective defined over token-level disentangled note properties, which enables a single generative model to adapt through lightweight fine-tuning.
- To support effective learning under the proposed objective, we introduce an **efficient and modeling-friendly tokenization scheme** for multitrack music that produces shorter sequences with lower complexity, facilitates instrument-level control and modeling that are important for arrangement performance. It further reduces note-level perplexity in unconditional generation, indicating potential utility beyond arrangement tasks.
- Instantiated with a small model and modest-scale pre-training, our system **outperforms task-specific SOTA baselines** on three representative arrangement tasks, each corresponding to a distinct scenario—band arrangement (reinterpretation), piano reduction (simplification), and drum arrangement (additive generation)—in both objective and subjective evaluations.

2 Related Work

2.1 Automatic Arrangement in Multitrack Symbolic Music

Symbolic music arrangement encompasses a variety of tasks such as chord progression generation [38], orchestration from lead sheets [34], and instrumentation transfer [42]. In this work, we focus on adapting multitrack music to new instrumentations—a representative setting that requires fine-grained control over musical content and instrumentation.

Early supervised learning approaches rely on parallel datasets (e.g., piano-to-orchestra [5], band-to-piano [29, 30]), which are expensive to construct and inherently constrain the direction of arrangement. Classification-based methods [9] approach the arrangement problem by learning to assign instrument labels to each note from a dataset with fixed instrumentation. However, such models are limited in expressiveness, as they cannot modify musical content—e.g., adding, removing, or altering notes—to suit a target instrument combination. Moreover, their reliance on fixed instrument sets hinders generalization to unseen combinations.

Recent self-supervised methods [41, 42] avoid these constraints by guiding generation with pre-defined or autoregressively modeled high-level descriptors (e.g., pitch histogram, note density). While effective for maintaining stylistic coherence, they separate style modeling from content realization. As a result, instrument playing styles are either fixed or modeled independently of input variations. Since musical material often changes between sections, this decoupling can degrade fidelity—causing arrangements to fail to reflect core aspects such as melody or texture, resulting in noticeable perceptual divergence. This suggests a more fidelity-oriented approach: integrating style modeling into the generation process to allow dynamic adaptation to input content.

Additionally, infilling-based models such as Composer’s Assistant [20, 19], although capable of handling additive generation scenarios such as drum arrangement, do not preserve the music essence of the original composition and are thus unsuited for reinterpretation and simplification.

2.2 Symbolic Music Tokenization

Transformer-based symbolic music modeling requires converting musical data into token sequences, a nontrivial task due to music’s inherent multi-stream structure, i.e., multiple instruments playing concurrently. ABC-based notations [23] convert staff-based music into text-like symbolic representations, which are well suited for classical sheet music. While for contemporary music, many existing tokenization schemes operate on MIDI files and adopt a linearized, note-event-based encoding where each musical note is represented as a tuple of attribute tokens (e.g., onset, pitch, duration, velocity, instrument). Some use absolute timing [13, 12, 39, 10], while others use metric durations [26, 14, 31]. Among them, the REMI representation [14], originally designed for single-track music, has been extended to REMI+ [31] for flexible tokenization of multitrack music, by associating each note with an instrument token.

However, existing time-ordered tokenization schemes such as REMI+ face structural limitations that hinder track-wise control and modeling. By flattening multitrack music into a strictly time-ordered sequence, REMI+ interleaves events from different instruments, resulting in **content fragmentation**. As illustrated in Figure 3a, notes from the same instrument (e.g., i-29, distorted electric guitar, in orange) are interrupted by those from concurrent instruments (e.g., i-80, synth lead, in red). This leads to two major issues. First, the lack of structured syntax makes it difficult to delineate track boundaries, hindering enforcement of user-specified instrument constraints and often leads to spurious instruments in arrangement outputs. Second, identical track-wise content can be tokenized differently depending on concurrent instrument activity. This context sensitivity reduces the repetition of typical per-instrument patterns in the training data—a key factor for learning accurate “instrument syntax” determined by physical constraints and idiomatic playing patterns of each instrument, important for arrangement quality. While some recent work explores vocabulary-level compression (e.g., Byte Pair Encoding [11]) to reduce sequence length—an approach orthogonal and complementary to tokenization scheme design—it does not address the structural issues discussed above.

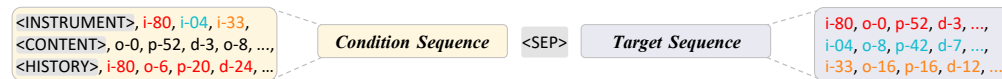


Figure 2: An example of the tokenized sequence for the band arrangement task. Special tokens $\langle \text{SEP} \rangle$, $\langle \text{INSTRUMENT} \rangle$, $\langle \text{CONTENT} \rangle$, and $\langle \text{HISTORY} \rangle$ are used to separate different components. Tokens starting with o-, i-, p-, d- represents the onset, instrument ID, pitch, and duration of notes.

3 Method

Our goal is to fine-tune a single pre-trained symbolic music generative model across diverse arrangement tasks via a unified pipeline. The core is a segment-level reconstruction objective (§3.1) over token-level disentangled representations of style (instrumentation) and content (musical notes). To support track-wise control and modeling, we introduce a structured tokenization scheme (§3.2) that reduces content fragmentation and enhances learning under this objective.

3.1 Reconstruction from Token-Level Disentangled Multi-Streams with Context Awareness

Symbolic music offers a unique opportunity for token-level disentanglement: each note is represented as a list of semantically independent tokens, each describing a unique property of the note (e.g., onset, duration, pitch, instrument), allowing content (what is played) and instrumentation (by whom it is played) to be explicitly separated. This level of structural redundancy is rare in natural language, where sub-word tokens are atomic.

Building on this observation, we formulate arrangement as a self-supervised, segment-level reconstruction task. The input music is decomposed into three token streams—instrument, content, and preceding context—and a pre-trained symbolic music generative model is fine-tuned to reconstruct the desired tracks of the multitrack music from these components.

Let $y^{(t)}$ denote the t -th *segment* of a music piece. The fine-tuning objective is defined as:

$$\mathcal{L}(\theta) = -\log p_{\theta}(\mathcal{T}_{\text{task}}(y^{(t)}) \mid \mathbf{I}(\mathcal{T}_{\text{task}}(y^{(t)})), \mathbf{C}(\mathcal{S}_{\text{task}}(y^{(t)})), \mathcal{T}_{\text{task}}(y^{(t-1)})), \quad (1)$$

where θ represents the model parameters, $\mathbf{I}(\cdot)$ and $\mathbf{C}(\cdot)$ extract instrument and content conditions, $\mathcal{S}_{\text{task}}$ and $\mathcal{T}_{\text{task}}$ are filters that select task-specific source and target tracks respectively (detailed in §3.3), and $y^{(t-1)}$ provides target-side history. Equation 1 is implemented using a standard next-token prediction objective on the sequence [condition] $\langle \text{sep} \rangle$ [target] (Figure 2), with cross-entropy loss computed only on the target subsequence.

Instrument condition. The instrument condition specifies the desired instruments to be used in the segment. During training, it includes all instrument tokens from the target sequence. Their order encodes pitch register relationships across tracks: instruments with higher average pitch are placed earlier. At inference, users can specify arbitrary desired instruments (instrument control) and define their relative pitch register ordering (voice control).

Content condition. The content condition is derived from the original composition, encoded as a *content sequence* without instrument information. To obtain it, we remove all instrument tokens from the original multitrack sequence, sort notes by onset time, then by descending pitch, and merge duplicates. This yields a clean time-ordered note sequence conveying what is played, independent of how or by whom.

History condition. The history condition provides musical context from the preceding segment, encouraging inter-segment coherence at inference time. During training, it is provided via teacher-forcing with the complete tokenized previous segment; at inference, it is autoregressively generated. This mechanism helps maintain coherent arrangement style across segments, especially important for long-form (e.g., song-level) arrangement scenarios.

During training, the instrument, content, and history conditions are derived from the same music being reconstructed. The model thus learns to reinterpret musical content with various instrument combinations under specific contexts, enabling diverse arrangement behaviors.

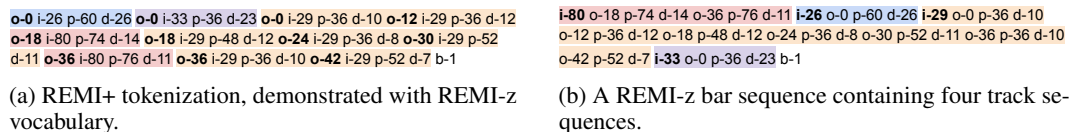


Figure 3: REMI+ and REMI-z tokenization for the same bar. Contents of the same instruments are highlighted with the same color. See Appendix A for complete vocabulary.

3.2 REMI-z: Tokenizing Multitrack Music with Track-Wise Continuity

As discussed in Section 2.2, strictly time-ordered tokenization schemes suffer from content fragmentation, a structural limitation that hinders arrangement performance. To mitigate this issue, we propose *REMI-z*, a tokenization scheme that heuristically prioritizes track-wise continuity over global time ordering. Specifically, REMI-z processes MIDI files into a list of *bar sequences*, each containing multiple *track sequences*, where each track corresponds to a unique instrument. Within each track, note events are sorted by their onset position, then by descending pitch, and grouped under a single instrument token. Tracks within a bar are then ordered by their average pitch (high to low), forming a bar-level token sequence that ends with a special end-of-bar token. The complete vocabulary and tokenization examples are shown in Appendix A.

This zig-zag organization—reflected in the REMI-z name—offers several modeling advantages: (1) instrument-wise content is locally contiguous (Figure 3b), reducing content fragmentation and enabling clear track boundaries; (2) sequence length is reduced by eliminating redundant instrument tokens; (3) temporal structure is preserved both within individual track sequences and between bars.

We adopt REMI-z to tokenize all data for both pre-training and fine-tuning. Beyond improving arrangement performance, we also observe that REMI-z produces sequences with lower information entropy and, when used for unconditional generative training, leads to better note-level modeling (see § 5.4). This suggests that its track-continuity design not only benefits arrangement tasks, but may also support general symbolic music modeling.

3.3 Task Instantiations

We evaluate our method on three representative music arrangement tasks that reflect typical arrangement scenarios, each assessing different capabilities of the model: band arrangement (reinterpretation), piano reduction (simplification), and drum arrangement (additive generation).

Band Arrangement. This task evaluates the model’s ability to reinterpret an existing piece using arbitrary combinations of pitched instruments. The model must learn the properties and idiomatic playing styles of various instruments to reallocate or generate notes appropriately. Both $\mathcal{S}_{\text{task}}$ and $\mathcal{T}_{\text{task}}$ are identity mappings (i.e., $\mathcal{S}_{\text{task}}(y) = \mathcal{T}_{\text{task}}(y) = y$), and drum tracks are removed from the input. To encourage creative rewriting, we randomly remove a subset of tracks from the content condition $C(\mathcal{S}_{\text{task}}(y^{(t)}))$ during training, while ensuring the melody’s content is preserved (detailed in Appendix B.2). Duration tokens are also removed from the content stream, allowing the model to infer track-specific note durations, i.e., articulations, suitable for interpreting the content under desired instrumentations. Segment length is set to 1 bar. This setup resembles [42], but differs in its more challenging setting: no track-wise style priors are available to guide the generation process.

Piano Reduction. This task simplifies ensemble music into solo piano accompaniment, aiming to preserve key harmonic and textural elements while ensuring pianistic playability. The $\mathcal{T}_{\text{task}}$ selects piano tracks, while $\mathcal{S}_{\text{task}}$ is the identity mapping. To ensure the reduction is meaningful, we filter training data to keep only segments where the piano part is sufficiently prominent (covering >40% of the pitch range). Drum tracks are removed from input, and the segment length is set to 1 bar. The setup is similar to [30], but uses original MIDI piano tracks as targets instead of human-composed reductions, and does not include difficulty-level conditioning.

Drum Arrangement. The goal of this task is to generate a drum track for music that lacks one. Here, $\mathcal{S}_{\text{task}}$ extracts all pitched-instrument tracks, while $\mathcal{T}_{\text{task}}$ extracts the drum track. We use 4-bar segments since drum patterns often span across multiple bars. The model must recognize the underlying groove and phrase boundaries of the source music, enhance them with coherent and stylistically appropriate rhythmic patterns, and ensure proper transitions across segments—requiring a stronger understanding

of rhythmic ideas and structural organization. This track infilling setup is similar to [19], but the model is not allowed to condition on future segments.

4 Experiments

4.1 Implementation Details

Our model, an 80M-parameter decoder-only Transformer, has a hidden dimension of 768, 12 layers, 16-head attention, and a context length of 2048 tokens (around $8\times$ the longest bar in our dataset). The model first undergoes a standard next-token-prediction pre-training, and then was fine-tuned with the proposed objective. Pre-training used four RTX A5000 GPUs (batch size 12, 1 epoch), while fine-tuning used a single A40 GPU (variable batch size, 3 epochs). Pre-training adopted the Los Angeles MIDI dataset [15] (405K MIDI files, 4.3B tokens after REMI-z tokenization, 2% validation split) and fine-tuning was done with Slakh2100 [21] (1,289 training, 270 validation, 151 test MIDI files), featuring 34 pitched instruments and drums, with ≥ 4 tracks per piece. Detailed hyperparameter settings are in Appendix B.5.

4.2 Baseline Models

For each task, we compare our model against a state-of-the-art (SOTA) task-specific baseline. For band arrangement, we adopt **Transformer-VAE** from [42], the strongest previously reported model for multitrack arrangement without assumptions on track type or number. It combines Transformer-based long-term and inter-track modeling with a VQ-VAE generation module. For piano reduction, we compare with [30] (**UNet**), the most recent work in this area. For drum arrangement, we adopt Composer’s Assistant 2 (**CA v2**) [19], a SOTA track infilling model capable of handling multitrack inputs and generating drum outputs. Existing drum-specific models (e.g., [2], [6]) are unsuitable for our setting, as they assume a single melody or instrumental track input rather than general multitrack conditioning. Baseline’s implementation details are in Appendix B.4.

To demonstrate the impact of generative pre-training, an ablation variant of our model without pre-training (**w/o PT**) is used as a baseline. Additionally, we include simple rule-based baselines as non-learning references, designed to provide naive solutions with minimal musical knowledge, helping to contextualize the difficulty of arrangement tasks. For band arrangement, **Rule-Based** distributes notes evenly by pitch across instruments, serving as a naive reinterpretation strategy. For piano reduction, we use **Rule-F** (a flattened multitrack where the piano plays all notes), which reflects an overcomplete reduction prioritizing coverage, and **Rule-O** (the original piano track), which provides a playability-guaranteed but musically incomplete reduction. For drum arrangement, the original drum track (**Ground Truth**) is included anonymously in the human evaluation as an upper bound on perceptual scores.

4.3 Objective Evaluation

Objective metrics measure similarity between model outputs and target sequences, assuming closer resemblance to human-created music indicates higher naturalness and musicality. Following [19, 30], we use *note-level F1* to measure similarity between model outputs and target sequences. Specifically, we compute **Note F1** (correct onset and pitch) and **Note_i F1** (additional correct instrument prediction), both under 16th-note quantization for fair comparison with baselines.

For piano reduction and drum arrangement, the same models are used in objective and subjective evaluations. For band arrangement, models are separately trained without random track deletion to ensure deterministic outputs. Baseline models are also modified by excluding its prior model to remove long-term context hints, ensuring evaluation fairness.

In addition to the modifications described above, we introduce three task-specific metrics for band arrangement. First, **Instrument Intersection over Union (I-IoU)** evaluates the accuracy of instrument control. Second, **Voice Error Rate (VER)** measures the similarity in voice features between the generated output and the reference, reflecting how well the model follows the voice conditions specified by instrument token ordering. Third, **Melody F1 (Mel F1)** computes the Note F1 score on melody tracks, estimated as the tracks with the highest average pitch in the output and reference, to

Table 1: Objective evaluation results for the band arrangement task. Statistical significance is indicated as follows: * for $p < 0.05$, † for $p < 0.01$, and ‡ for $p < 0.001$.

Model	I-IOU ↑	VER ↓	Note F1 ↑	Note _i F1 ↑	Mel F1 ↑
Transformer-VAE	97.5	35.0	49.5	40.0	24.7
Transformer w/ REMI+	95.0	18.2	94.4	76.0	68.8
Transformer w/ REMI-z	‡99.5	‡9.9	‡ 97.8	‡77.5	‡77.8
+ Pre-training (Ours)	99.8	7.6	97.5	87.0	84.5
– voice	99.6	17.6	97.2	84.3	81.5
– history	100.0	9.0	97.6	77.4	79.4

assess how well the original melody is preserved—an important factor for perceived fidelity. Detailed definitions and computation procedures are provided in Appendix C.1.

Among all tasks, band arrangement is a strong testbed for evaluating controllability and generalizability because it requires the highest flexibility without assumptions on target instrument types or counts. Hence it serves two additional purposes: (1) to validate key design choices in our fine-tuning objective, particularly the use of voice-aware instrumentation and segment-level history conditioning; and (2) to prove the effectiveness of the proposed tokenization schemes in arrangement task. When comparing tokenization schemes, statistical significance is computed by Wilcoxon signed rank test [36]. For a broader analysis of tokenization’s impact on unconditional generation, see § 5.4.

4.4 Human Evaluation

To complement similarity-based objective metrics and assess perceptual quality and creativity, we conducted human evaluations. Full-piece arrangements were generated by all models and evaluated on a 5-point scale (1: very low, 5: very high). For band arrangement, models were tested across three instrument combinations with different complexity: string trio (3 tracks), rock band (4 tracks), and jazz band (7 tracks). Three metrics were used across band, piano, and drum arrangement tasks: **Coherence**, which evaluates the natural flow of the arrangement and the consistency of each instrument’s playing style throughout the piece; **Creativity**, which assesses the degree of innovation in the arrangement under the constraints of the music’s content and style; and **Musicality**, which measures the overall musical appeal and aesthetic quality of the arrangement. Further details on the metrics, ensemble settings, questionnaire, and evaluation process are in Appendix D.

Task-specific metrics were introduced for the distinct evaluation needs of each arrangement scenario. For band arrangement, **Faithfulness** measures resemblance to the original in melody and overall feel, while **Instrumentation** assesses the appropriateness of each instrument’s role within the ensemble and their harmony. For piano reduction, **Faithfulness** is also adopted but without melody preservation requirements, and **Playability** assesses the feasibility of the generated contents played by human pianists. For drum arrangement, **Compatibility** measures how well the drum track blends with other instruments, and **Phrase Transition** assesses the smoothness of transitions between musical phrases. We report mean and standard error of mean in result tables. Significance tests were conducted between our model and the SOTA baselines using within-subject (repeated-measures) ANOVA [28].

5 Results

5.1 Band Arrangement

5.1.1 Objective Evaluation

Tokenization impact. Table 1 shows that model adopted REMI-z significantly outperforms that adopted REMI+ across all objective metrics. It enables stronger instrument control (I-IOU: 99.5% vs. 95.0%) and voice control (VER: 9.9% vs. 18.2%), aligning outputs better with user-specified conditions. The improvement in Note F1 (+6.7%) confirms higher reconstruction quality, while the gain in Note_i F1 (+1.9%) highlights enhanced instrument-wise modeling. Furthermore, the substantial increase in Mel F1 (+9.0%) suggests REMI-z enable the model to better identify the melody components from content sequence, which is important for arrangement fidelity.

Table 2: Band arrangement subjective evaluation results. Fa., Co., In., Cr., and Mu. represent Faithfulness, Coherence, Instrumentation, Creativity, and Musicality, respectively.

Model	Fa. $\uparrow$	Co. $\uparrow$	In. $\uparrow$	Cr. $\uparrow$	Mu. $\uparrow$
Rule-Based	3.46 $\pm$ 0.14	3.05 $\pm$ 0.13	2.89 $\pm$ 0.14	3.00 $\pm$ 0.12	3.07 $\pm$ 0.13
Transformer-VAE	2.65 $\pm$ 0.10	2.70 $\pm$ 0.11	2.72 $\pm$ 0.12	3.00 $\pm$ 0.13	2.72 $\pm$ 0.11
Ours	$\ddagger$ 3.77 $\pm$ 0.13	$\ddagger$ 3.47 $\pm$ 0.15	$\ddagger$ 3.49 $\pm$ 0.16	*3.40 $\pm$ 0.13	$\ddagger$ 3.47 $\pm$ 0.14
w/o PT	3.19 $\pm$ 0.13	2.82 $\pm$ 0.13	2.86 $\pm$ 0.14	2.93 $\pm$ 0.12	2.75 $\pm$ 0.13

Table 3: Piano reduction results. The Pl. represents Playability score.

Model	F1 $\uparrow$	Fa. $\uparrow$	Co. $\uparrow$	Pl. $\uparrow$	Cr. $\uparrow$	Mu. $\uparrow$
Rule-F	-	3.93 $\pm$ 0.13	3.59 $\pm$ 0.13	3.14 $\pm$ 0.13	2.96 $\pm$ 0.13	3.34 $\pm$ 0.14
Rule-O	-	2.75 $\pm$ 0.13	3.49 $\pm$ 0.13	4.07 $\pm$ 0.12	2.62 $\pm$ 0.14	2.96 $\pm$ 0.14
UNet	58.3	2.97 $\pm$ 0.13	2.90 $\pm$ 0.15	3.47 $\pm$ 0.13	2.82 $\pm$ 0.13	2.78 $\pm$ 0.13
Ours	85.5	$\ddagger$ 3.63 $\pm$ 0.13	$\ddagger$ 3.64 $\pm$ 0.13	*3.86 $\pm$ 0.13	*3.14 $\pm$ 0.12	$\ddagger$ 3.48 $\pm$ 0.14
w/o PT	78.4	2.25 $\pm$ 0.13	2.58 $\pm$ 0.16	3.29 $\pm$ 0.15	2.67 $\pm$ 0.15	2.26 $\pm$ 0.14

Pre-training benefit. Pre-training brings clear gains by transferring musical knowledge useful for arrangement. It improves three key aspects: (1) lower VER (-2.3%), indicating more effective voice control; (2) higher Note_i F1 ($+9.5\%$), reflecting enhanced instrument-wise content modeling; and (3) the highest Mel F1 (84.5%), indicating stronger melody preservation. Probing analysis (Appendix E) further shows that pre-training strengthens the alignment between token embeddings and musical concepts useful for arrangement, such as instrument roles and chord progression.

Comparison with Transformer-VAE. Our model outperforms Transformer-VAE by a wide margin. In particular, it achieves much higher Note F1 (97.5% vs. 49.5%), Note_i F1 (87.0% vs. 40.0%), and Mel F1 (84.5% vs. 24.7%), highlighting the advantage of our context-aware, content- and instrument-conditioned generation approach over the latent inference used in Transformer-VAE.

Ablation: voice and history conditioning. Removing voice information from instrument conditions degrades voice control (VER: $+10.0\%$), demonstrating the effectiveness of our voice control method. It also lowers per-instrument F1 (Note_i: -2.7%) and Mel F1 (-3.0%), indicating that voice-order information serves as a useful hint for inferring instrument roles during arrangement. Excluding history conditioning results in even larger drops in Note_i F1 (-9.6%) and Mel F1 (-5.1%), validating that temporal context facilitates accurate reconstruction, laying the foundation for coherent song-level arrangement.

5.1.2 Human Evaluation

Strong subjective performance. As shown in Table 2, Our model achieves the highest scores across all subjective metrics than all baselines, demonstrating its ability to generate coherent, stylistically appropriate, and musically appealing arrangements while preserving core musical essence.

Transformer-VAE underperforms. It lags significantly behind our model in every subjective criterion. Notably, its lower Faithfulness score (-1.12) reflects difficulty in preserving core musical content, while low ratings in Instrumentation (-0.77) and Coherence (-0.77) suggest weaker track-wise modeling and non-idiomatic instrument usage, as well as insufficient coherence between segments. Overall Musicality (-0.75) also falls below our model. It even scores lower than the rule-based baseline, though the latter suffers from inherent limitations in musicality and coherence.

Pre-training improves quality. Removing pre-training reduces scores across all metrics, with notable drops in Faithfulness (-0.58) and Musicality (-0.72), reinforcing the importance of musical knowledge transfer from generative pre-training for content retention and overall perceptual quality.

5.2 Piano Reduction

Best overall quality. As shown in Table 3, our method obtains the highest ratings in F1 (85.5%), Coherence (3.64), Playability (3.86), and Musicality (3.48), while also maintaining strong scores in Faithfulness (3.63) and Creativity (3.14). It significantly outperforms UNet across all subjective

Table 4: Drum arrangement results. Comp. and Tr. represent Compatibility and Phrase Transition score respectively.

Model	F1 $\uparrow$	Comp. $\uparrow$	Co. $\uparrow$	Tr. $\uparrow$	Cr. $\uparrow$	Mu. $\uparrow$
Ground Truth	100.0	4.31 ± 0.12	4.18 ± 0.10	3.36 ± 0.13	3.16 ± 0.12	3.78 ± 0.12
CA v2	20.3	3.82 ± 0.13	4.05 ± 0.12	2.86 ± 0.12	2.58 ± 0.11	3.19 ± 0.12
Ours	79.3	3.91 ± 0.12	4.03 ± 0.10	$\ddagger$ 3.77 ± 0.12	$\ddagger$ 3.27 ± 0.14	$\ddagger$ 3.57 ± 0.13
w/o PT	1.2	2.49 ± 0.16	2.19 ± 0.12	2.21 ± 0.14	2.82 ± 0.15	2.05 ± 0.13

Table 5: Tokenization scheme comparison on unconditional generation.

Tokenizer	$\bar{T}_{\text{bar}} \downarrow$	$\bar{T}_{\text{note}} \downarrow$	$\bar{H}_{\text{bar}} \downarrow$	$\text{PPL}_{\text{note}} \downarrow$	$\text{PPL}_{\text{token}} \downarrow$
REMI+	225.91	4.03	41.68	116.20	3.00
REMI-z (Ours)	151.68	2.77	29.43	84.11	4.50

metrics, with especially large margins in Faithfulness (+0.66), Coherence (+0.74), and Musicality (+0.70)—indicating better content preservation, cross-segment continuity, and overall musical quality. Pre-training again proves essential: without it (w/o PT), performance drops notably across all metrics.

Balanced fidelity and playability. Compared to rule-based methods, our model achieves a better trade-off between fidelity and playability. Rule-F preserves original content well (Faithfulness: 3.93) but results in poor playability (3.14), while Rule-O achieves the highest playability (4.07) but at the expense of faithfulness (2.75). In contrast, our method produces reductions with balanced Faithfulness (3.63) and Playability (3.86).

5.3 Drum Arrangement

Best subjective quality among learned models. As shown in Table 4, our model outperforms CA v2 in all subjective metrics except Coherence (4.03 vs. 4.05), and significantly improves Creativity (3.47 vs. 2.58) and Phrase Transition (3.27 vs. 2.86), which are essential for engaging, structurally-aware drum arrangements. It also achieves the highest Musicality score (3.57), closely approaching ground truth (3.78). Again, the w/o PT variant performs poorly across all metrics.

Improved phrasing and variation. Compared to CA v2, which often repeats similar drum patterns throughout a piece, our model produces more varied and context-aware rhythms that better reflect musical phrasing. The higher Phrase Transition score (+0.91) supports this observation, demonstrating the model’s ability to capture structural changes in music during generation.

5.4 Tokenization Efficiency and General Modeling Advantages

Additionally, we evaluate REMI-z against REMI+ to assess its efficiency and effectiveness for generative modeling of symbolic multitrack music. We report several metrics to evaluate compactness of tokenization schemes and their unconditional modeling performance: 1) **average tokens per bar** ($\bar{T}_{\text{bar}}$), 2) **average tokens per note** ($\bar{T}_{\text{note}}$), 3) **Shannon entropy of bar-level token sequences** ($\bar{H}_{\text{bar}}$), and 4) **note-level perplexity** (PPL_{note}). Note that PPL_{note} is the aggregated conditional probability of all note attribute tokens, normalized by the number of notes, enabling fairer comparison across tokenization schemes. Calculations are detailed in Appendix C.2.

Compactness. Tokenizing Slakh2100 with REMI-z yields a 32.9% reduction in sequence length per bar (151.68 vs 225.91) and fewer tokens per note (2.77 vs 4.03), effectively reducing training and inference computational costs.

Representational Simplicity. REMI-z also produces lower bar-level Shannon entropy (29.43 vs 41.68 bits/token) on Slakh2100 compared to REMI+, despite representing the same underlying musical content. This indicates reduced information redundancy, and suggests that REMI-z sequences consist of more predictable tokens, potentially facilitating the learning of generative models.

Note-Level Modeling Advantage. We train unconditional generation models on Slakh2100 with both tokenizations and compare note-level perplexity with the same architecture as our arrangement models. The model adopting REMI-z achieves substantially lower note-level perplexity (84.11 vs.

116.20), indicating better modeling of complete musical notes. Since notes form the atomic units of music, improvements in note-level modeling directly contribute to lower uncertainty at modeling bars and full compositions, suggesting utility in general symbolic music modeling beyond arrangement tasks. While REMI+ slightly outperforms in token-level perplexity, this does not translate to better modeling of higher-level structures.

6 Conclusion

We presented a unified framework for automatic music arrangement, centered around a reconstruction objective that enables diverse arrangement tasks without requiring parallel data, enabling knowledge transfer from generative pre-trained symbolic music models for arrangement tasks. Our approach also integrates a structured tokenization scheme, REMI-z, which convert multitrack music to compact and easy-to-model token sequences. Experimental results on band arrangement, piano reduction, and drum arrangement show that our method consistently outperforms task-specific baselines in both objective and subjective evaluations. These results suggest the potential of our framework as a general solution for symbolic music-to-music transformation.

Acknowledgements

The authors would like to thank Chenfei Kang for providing valuable resources that supported the validation of the proposed idea, and Yisong Miao for insightful comments on training and coherence of language models, which substantially benefited this work.

References

- [1] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.
- [2] Giorgio Barnabò, Giovanni Trappolini, Lorenzo Lastilla, Cesare Campagnano, Angela Fan, Fabio Petroni, and Fabrizio Silvestri. Cycledrums: automatic drum arrangement for bass lines using cylegan. *Discover Artificial Intelligence*, 3(1):4, 2023.
- [3] Kristy Choi, Curtis Hawthorne, Ian Simon, Monica Dinculescu, and Jesse Engel. Encoding musical style with transformer autoencoders. In *International conference on machine learning*, pages 1899–1908. PMLR, 2020.
- [4] Vince Corozine. *Arranging music for the real world: classical and commercial aspects*. Mel Bay Publications, 2015.
- [5] Léopold Crestel and Philippe Esling. Live orchestral piano, a system for real-time orchestral music generation. *arXiv preprint arXiv:1609.01203*, 2016.
- [6] Rishabh Dahale. *Automatic Drum Accompaniment Generation from Melody*. PhD thesis, Indian Institute of Technology Bombay, 2022.
- [7] Roger B Dannenberg and Sukrit Mohan. Characterizing tempo change in musical performances. In *ICMC*, 2011.
- [8] Hao-Wen Dong, Ke Chen, Shlomo Dubnov, Julian McAuley, and Taylor Berg-Kirkpatrick. Multitrack music transformer. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.
- [9] Hao-Wen Dong, Chris Donahue, Taylor Berg-Kirkpatrick, and Julian McAuley. Towards automatic instrumentation by learning to separate parts in symbolic multitrack music. In *International Society for Music Information Retrieval Conference*, 2021.
- [10] Jeffrey Ens and Philippe Pasquier. Mmm : Exploring conditional multi-track music generation with the transformer. *ArXiv*, abs/2008.06048, 2020.

- [11] Nathan Fradet, Nicolas Gutowski, Fabien Chhel, and Jean-Pierre Briot. Byte pair encoding for symbolic music. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2001–2020, Singapore, December 2023. Association for Computational Linguistics.
- [12] Josh Gardner, Ian Simon, Ethan Manilow, Curtis Hawthorne, and Jesse Engel. Mt3: Multi-task multitrack music transcription. *ArXiv*, abs/2111.03017, 2021.
- [13] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam M. Shazeer, Ian Simon, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations*, 2018.
- [14] Yu-Siang Huang and Yi-Hsuan Yang. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1180–1188, 2020.
- [15] Aleksandr Lev. Los angeles midi dataset: Sota kilo-scale midi dataset for mir and music ai purposes. In *GitHub*, 2024.
- [16] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.
- [17] Peiling Lu, Xin Xu, Chenfei Kang, Botao Yu, Chengyi Xing, Xu Tan, and Jiang Bian. Musecoco: Generating symbolic music from text. *arXiv preprint arXiv:2306.00110*, 2023.
- [18] Dimos Makris, Kat R Agres, and Dorien Herremans. Generating lead sheets with affect: A novel conditional seq2seq framework. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.
- [19] Martin Malandro. Composer’s Assistant 2: Interactive Multi-Track MIDI Infilling with Fine-Grained User Control. In *Proc. 25th Int. Society for Music Information Retrieval Conf.*, San Francisco, CA, USA, 2024.
- [20] Martin E Malandro. Composer’s assistant: An interactive transformer for multi-track midi infilling. *arXiv preprint arXiv:2301.12525*, 2023.
- [21] Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le Roux. Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 45–49. IEEE, 2019.
- [22] Philippe Pasquier, Jeff Ens, Nathan Fradet, Paul Triana, Davide Rizzotti, Jean-Baptiste Rolland, and Maryam Safi. Midi-gpt: A controllable generative model for computer-assisted multitrack music composition. *arXiv preprint arXiv:2501.17011*, 2025.
- [23] Xingwei Qu, Yuelin Bai, Yinghao Ma, Ziya Zhou, Ka Man Lo, Jiaheng Liu, Ruibin Yuan, Lejun Min, Xueling Liu, Tianyu Zhang, et al. Mupt: A generative symbolic music pretrained transformer. *arXiv preprint arXiv:2404.06393*, 2024.
- [24] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.
- [25] Colin Raffel. *Learning-Based Methods for Comparing Sequences, with Applications to Audio-to-MIDI Alignment and Matching*. Phd thesis, Columbia University, USA, 2016.
- [26] Yi Ren, Jinzheng He, Xu Tan, Tao Qin, Zhou Zhao, and Tie-Yan Liu. Popmag: Pop music accompaniment generation. In *Proceedings of the 28th ACM international conference on multimedia*, pages 1198–1206, 2020.
- [27] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

- [28] Henry Scheffe. *The analysis of variance*, volume 72. John Wiley & Sons, 1999.
- [29] Moyu Terao, Yuki Hiramatsu, Ryoto Ishizuka, Yiming Wu, and Kazuyoshi Yoshii. Difficulty-aware neural band-to-piano score arrangement based on note-and statistic-level criteria. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 196–200. IEEE, 2022.
- [30] Moyu Terao, Eita Nakamura, and Kazuyoshi Yoshii. Neural band-to-piano score arrangement with stepless difficulty control. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.
- [31] Dimitri von Rütte, Luca Biggio, Yannic Kilcher, and Thomas Hofmann. Figaro: Controllable music generation using learned and expert features. In *The Eleventh International Conference on Learning Representations*, 2023.
- [32] Yashan Wang, Shangda Wu, Jianhuai Hu, Xingjian Du, Yueqi Peng, Yongxin Huang, Shuai Fan, Xiaobing Li, Feng Yu, and Maosong Sun. Notagen: Advancing musicality in symbolic music generation with large language model training paradigms. *arXiv preprint arXiv:2502.18008*, 2025.
- [33] Ziyu Wang, Ke Chen, Junyan Jiang, Yiyi Zhang, Maoran Xu, Shuqi Dai, Xianbin Gu, and Gus Xia. Pop909: A pop-song dataset for music arrangement generation. *arXiv preprint arXiv:2008.07142*, 2020.
- [34] Ziyu Wang, Lejun Min, and Gus Xia. Whole-song hierarchical generation of symbolic music using cascaded diffusion models. *arXiv preprint arXiv:2405.09901*, 2024.
- [35] Ziyu Wang, Dejing Xu, Gus Xia, and Ying Shan. Audio-to-symbolic arrangement via cross-modal music representation learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 181–185. IEEE, 2022.
- [36] Frank. Wilcoxon. Individual comparisons by ranking methods. *Biometrics*, 1:196–202, 1945.
- [37] Shangda Wu, Yashan Wang, Xiaobing Li, Feng Yu, and Maosong Sun. Melodyt5: A unified score-to-score transformer for symbolic music processing. *arXiv preprint arXiv:2407.02277*, 2024.
- [38] Li Yi, Haochen Hu, Jingwei Zhao, and Gus Xia. Accomontage2: A complete harmonization and accompaniment arrangement system. *arXiv preprint arXiv:2209.00353*, 2022.
- [39] Mingliang Zeng, Xu Tan, Rui Wang, Zeqian Ju, Tao Qin, and Tie-Yan Liu. Musicbert: Symbolic music understanding with large-scale pre-training. *ArXiv*, abs/2106.05630, 2021.
- [40] Xueyao Zhang, Jinchao Zhang, Yao Qiu, Li Wang, and Jie Zhou. Structure-enhanced pop music generation via harmony-aware learning. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1204–1213, 2022.
- [41] Jingwei Zhao, Gus Xia, and Ye Wang. Q&a: query-based representation learning for multi-track symbolic music re-arrangement. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 5878–5886, 2023.
- [42] Jingwei Zhao, Gus Xia, Ziyu Wang, and Ye Wang. Structured Multi-Track Accompaniment Arrangement via Style Prior Modelling. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

A REMI-z Tokenization

A.1 Vocabulary

Table 6 presents the complete vocabulary of our proposed REMI-z tokenizer, detailing the value ranges for each token type. For instrument tokens, the values correspond to MIDI program numbers (0–127), with 128 specifically designated for the drum set. For pitched instruments, pitch token values directly map to MIDI pitch numbers, while drum set pitches are encoded as MIDI pitch + 128. Both position and duration tokens are quantized in units of a 48th note (one-third of a sixteenth note). Time signature and tempo tokens undergo specific quantization before token mapping; for detailed quantization rules, please refer to our code implementation. All tokens listed in the table were utilized during pre-training, including time signature and tempo tokens serving as bar-level properties, with example REMI-z sequences illustrated in Figure 4b.

During fine-tuning, we simplified the token set by excluding time signature and tempo tokens. This decision was supported by our analysis of the fine-tuning dataset (Slakh2100 [21]), where 94.8% of songs use 4/4 time signatures. Consequently, we restricted our fine-tuning to 4/4 songs and omitted time signature tokens, following practices in related works [35, 42]. We also excluded tempo tokens since tempo adjustments in digital audio workstations are typically handled as a global parameter, affecting only tempo tokens without altering the any other tokens. This simplification assumes that compositional and performance styles remain consistent across different tempos. However, this assumption may not hold for datasets with significant tempo variations. Therefore, for future research requiring time-signature- or tempo-specific characteristics, we recommend including these tokens during fine-tuning.

Velocity tokens were excluded from both pre-training and fine-tuning phases. This decision reflects our focus on compositional quality rather than performance naturalism, aligning with previous approaches in band and piano arrangement studies [42, 30]. However, if velocity information is deemed crucial for generation, our model architecture readily accommodates the addition of velocity tokens to each note without introducing content fragmentation issues, if the REMI-z note organization order is maintained.

A.2 Example

We name our tokenization scheme *REMI-z* for its distinctive “zig-zag” encoding pattern for musical notes within each bar, as illustrated in Figure 4a. In this scheme, notes are encoded hierarchically: first grouped by tracks (instruments), then organized bar by bar. This track-first approach ensures notes from the same instrument remain clustered together, thereby enhancing the model’s ability to learn instrument-specific patterns. The resulting REMI-z sequence is demonstrated in Figure 4b. In contrast, REMI+ [31] employs a column-wise encoding strategy, strictly ordering notes by their temporal positions. While this approach effectively captures global temporal relationships, it disperses notes from the same instrument throughout the sequence, potentially complicating instrument-specific pattern learning. We leverage such temporal ordering in our content sequence (excluding instrument tokens), as shown in Figure 5.

Table 6: REMI-z vocabulary with pitch token distinctions

Meaning	Token	X’s range
Instrument type	i-X	0~128
Note’s within-bar position	o-X	0~127
Note’s pitch (non-drum)	p-X	0~127
Note’s pitch (drum)	p-X	128~255
Note’s duration	d-X	0~127
End of a bar	b-1	-
Time signature	s-X	0~253
Tempo	t-X	0~48



(a) A 4-bar musical segment with orange arrows illustrating the “zig-zag” encoding order of notes in REMI-z tokenization.



(b) The REMI-z sequence tokenized from the musical segment above. Track sequences are color-coded by instrument: synth lead (red), clean electric guitar (blue), distorted electric guitar (orange), and electric bass (purple).

Figure 4: An example of REMI-z tokenization.

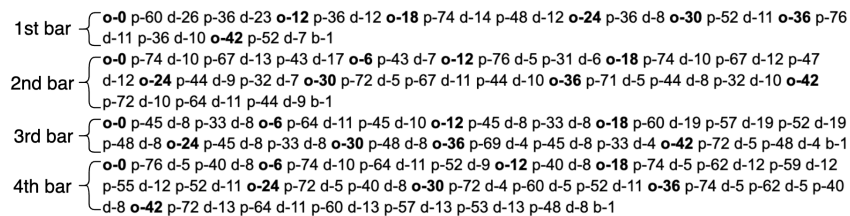


Figure 5: The content sequence obtained by applying the operator $C(\cdot)$ to the REMI-z sequence shown in Figure 4b.

B Implementation Details

B.1 Key Normalization

Transposition-equivariance is a crucial property in symbolic music: a composition’s musicality remains unchanged under global pitch shifts (uniform pitch adjustment across all notes). However, uneven key distribution in datasets can lead to data sparsity during training, potentially causing models to perform inconsistently across different keys. Two approaches address this issue: (1) data augmentation through systematic semitone transpositions $0, \pm 1, \pm 2, \dots$ [19, 42], or (2) normalizing all songs to a common key (e.g., C major and A minor) [17]. We adopt the latter approach, implementing a modified version of [17]’s method. Our implementation uses a key dictionary mapping from each of the 24 keys (12 major and 12 minor) to a 12-dimensional binary vectors, where 1s indicate scale notes. Key detection is performed by computing dot products between these vectors and a song’s pitch histogram, with the highest-scoring key determining the transposition needed to normalize to C major or A minor. Further details can be found in the code.

B.2 Random Track Deletion

To encourage creativity in band arrangement and prevent the model from over-relying on direct copying notes from the input, we introduce a random track deletion mechanism during training. The intuition is that a well-trained model should be able to infer suitable instrumental content even if such content does not exist in the original music composition.

Concretely, given a content sequence with multiple instrument tracks, we randomly delete all notes that belongs to a subset of instruments before calculating the content sequences and feeding to the model. The number of instruments to delete is sampled from a Poisson distribution with $\lambda = \max(\lfloor |\mathcal{I}|/4 \rfloor, 1)$, where $\mathcal{I}$ is the set of non-melodic instruments in the content sequence. The melodic track is estimated by the track with highest average pitch, and does not involves in the track deletion to ensure the melody's content is preserved. Then, the deletion count is clipped to ensure at least one instrument remains. A corresponding number of instruments are uniformly sampled without replacement and all tokens associated with these instruments are removed from the input. Then the content sequence is calculated on this modified music sequence.

Importantly, the target sequence remains unchanged: it includes all instruments and notes removed from the input. This setup requires the model to reconstruct missing tracks based solely on the remaining musical context. In doing so, the model learns not only to replicate observed content but also to infer plausible notes for desired instruments in a given context. This can be treated as a music specific span infilling denoising objective for sequence-to-sequence learning [16], but we don't explicitly tell the model whether an input is masked and the location of masked spans.

B.3 Instrument Quantization

Band Arrangement While our tokenization scheme supports all MIDI program IDs, many instruments with different IDs share fundamental compositional properties, differing primarily in timbre (e.g., acoustic and electric pianos). To leverage these similarities and reduce instrument distribution sparsity in training data, during fine-tuning, we group similar instruments and assign them the lowest program ID within their group. The instrument grouping rules follows that of [21], leading to 34 different program IDs in total. For multiple tracks of the same instrument type, we merge their notes into a single track.

Piano Arrangement For piano arrangement, we consolidate all piano-type instruments (MIDI program IDs 0-7, including both acoustic and electric pianos) into a single track to form the target sequence.

B.4 Baseline Models

Band Arrangement For band arrangement, we use the Transformer-VAE model from [42] as our baseline. For human evaluation, we utilized their official implementation without retraining. The model's generation module was trained on Slakh2100 [21] (the same dataset as our fine-tuning), and its prior model leveraged the larger Lakh MIDI Dataset [25], which encompasses Slakh2100. For objective evaluation, we retrained another version of the model on Slakh2100 that did not receive hints from the style prior model for a fair comparison.

Piano Reduction Piano reductions can be categorized into two types: (1) piano accompaniments where the melody is delegated to a separate lead instrument, focusing solely on preserving harmony and texture (e.g., [41, 33]), which can be effectively handled using self-supervised methods; and (2) complete solo piano arrangements that additionally include the melody, requiring careful human arrangement and band-to-piano parallel datasets for supervised learning (e.g., [30, 29]). Due to the lack of open-source parallel datasets, we focus on the accompaniment arrangement task in this paper and use the term *piano reduction* interchangeably. However, our methodology is inherently flexible and can also handle the second type of reduction if parallel datasets become available. Additionally, exploring the potential for full solo piano reductions using unsupervised approaches remains a valuable direction for future research.

For comparison, we reimplemented the UNet baseline from [30] and trained it on Slakh2100 using our data preparation pipeline. Instead of adopting parallel data, we followed the same setting as our model, using the original composition as input and the piano track within the song as output. To

Table 7: Pre-train hyper-parameter setting.

Hyperparameters	Values
seed	42
learning_rate	0.0005
weight_decay	0.1
train_batch_size	12
gradient_accumulation_steps	8
total_train_batch_size	96
optimizer	Adam with betas=(0.9,0.999) and epsilon=1e-08
lr_scheduler_type	cosine
lr_scheduler_warmup_steps	1000
num_epochs	1

ensure musical coverage, we selected only piano tracks that span more than 40% of the piece’s pitch range.

Since hand-specific annotations (i.e., left and right hand separation) are unavailable, the entire piano track is generated as a single sequence rather than as separate streams for each hand. We also omitted octave shifts during input preprocessing for two reasons. The first is that, unlike their setup where human-composed piano references may intentionally transpose notes by octaves, our output consistently corresponds to a subset of the input notes without such shifts. The second is empirical: introducing octave shifts during training significantly degraded UNet’s performance, reducing note-level F1 from 58.31 to 42.80 and diminishing perceptual quality.

Drum Arrangement For drum arrangement, we employ Composer’s Assistant v2 [19] as our baseline, specifically using v2.1.0 from their official repository². We use the model without retraining since its original training data (Lakh MIDI Dataset [25]) encompasses our fine-tuning dataset (Slakh2100).

Tokenization Comparison When comparing the effectiveness of REMI+ and the proposed REMI-z tokenization in §3.2, the model structure used is the same as our arrangement model, but with a simpler training objective—the standard left-to-right next-token-prediction. We did not use time signature tokens, tempo tokens, and velocity tokens in REMI+ for a fair comparison.

B.5 Hyperparameter Settings

The model we adopted is a GPT-2 model comprising 12 Transformer decoder layers with a hidden size of 768. Our setup follows GPT’s 12-layer, 768-dim embedding, 3072-dim inner states [24], and our experimental design is also in the same style: a single pretrained model and multiple fine-tuning tasks to show the efficacy of the pretrain–finetune paradigm. We used a slightly higher number of attention heads (16 instead of 12) based on the intuition that the interactions between different music tokens are more critical for musical quality than the value of individual note attributes. We detail the hyperparameter configurations used in our experiments below.

For pre-training, Table 7 summarizes the pre-training configuration, where hyperparameters were used as-is without optimization. The pre-training was implemented using `pytorch` and `transformers` frameworks on a Linux platform, while fine-tuning additionally utilized `lightning`.

For fine-tuning, we conducted a simple learning rate search over 1e-5, 5e-5, 1e-4, selecting the optimal value based on validation loss. This resulted in learning rates of 5e-5 for drum arrangement and 1e-4 for band arrangement and piano reduction. The batch sizes and context lengths were configured as follows: band arrangement used a batch size of 24 and context length of 768; piano reduction similarly adopted a batch size of 24 and context length of 768; drum arrangement employed a batch size of 8 and context length of 1536. Across all fine-tuning tasks, we used the AdamW optimizer with

²<https://github.com/m-malandro/composers-assistant-REAPER>

0.01 weight decay, incorporating a linear learning rate scheduler with 500-step warmup. Training spanned 3 epochs for band, piano, and drum tasks, with early stopping patience of 2 epochs. The best checkpoints were selected based on validation loss.

For the tokenization comparison, the model is trained from scratch without generative pre-training. The hyperparameter settings remain the same as those used for band arrangement fine-tuning, except for an increased number of epochs (5).

All experiments were conducted with a fixed random seed of 42.

C Objective Metrics

In this section, we formally define the objective metrics used in the paper. All objective metrics are calculated at the segment level, and arithmetically averaged across the test set.

C.1 Arrangement Evaluation

Most of our objective metrics are based on note-level F1 scores calculated on piano roll of 16-th note quantization. Given the extreme sparsity of note events in the track-wise piano roll (e.g., only 0.12% non-zero elements in the Slakh2100 dataset under 16th-note quantization), F1-based metrics are more suitable than accuracy-based metrics for similarity evaluation. Let $X = \{x_1, \dots, x_n\}$ and $Y = \{y_1, \dots, y_m\}$ be two sequences of note events, where each note event e is defined as a tuple $e = (t, p)$ with onset time t and pitch p . A note event $x_i \in X$ is considered to match $y_j \in Y$ if and only if:

$$|t_{x_i} - t_{y_j}| < \delta_t \text{ and } p_{x_i} = p_{y_j} \quad (2)$$

where δ_t is the temporal tolerance threshold (set to one 16th-note duration in our evaluation).

Let $M(X, Y)$ denote the set of matched note pairs between X and Y . The **Note F1** score is defined as:

$$\text{Precision} = \frac{|M(X, Y)|}{|X|} \quad (3)$$

$$\text{Recall} = \frac{|M(X, Y)|}{|Y|} \quad (4)$$

$$\text{Note F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (5)$$

The **Note_i F1** extends this by considering instrument matching, where each note event becomes $e = (t, p, i)$ with i representing the instrument. The matching criterion becomes:

$$|t_{x_i} - t_{y_j}| < \delta_t \text{ and } p_{x_i} = p_{y_j} \text{ and } i_{x_i} = i_{y_j} \quad (6)$$

Note_i F1 is then computed using Equation 5 with this stricter matching criterion.

For **Melody F1 (Mel F1)**, given a multitrack piece with tracks $T = \{T_1, \dots, T_k\}$, we first identify the melody track T_m as:

$$T_m = \arg \max_{T_i \in T} \frac{1}{|T_i|} \sum_{e \in T_i} p_e \quad (7)$$

where $|T_i|$ is the number of notes in track T_i and p_e is the pitch of note event e . The Mel F1 score between output and target sequences is then:

$$\text{Mel F1} = \text{Note F1}(T_m^{\text{out}}, T_m^{\text{tgt}}) \quad (8)$$

For instrument control evaluation, **Instrument IoU (I-IoU)** measures the overlap between instrument sets. Let $\mathcal{I}^{\text{out}}$ and $\mathcal{I}^{\text{tgt}}$ denote the sets of instruments used in the output and target sequences respectively:

$$\text{I-IoU} = \frac{|\mathcal{I}^{\text{out}} \cap \mathcal{I}^{\text{tgt}}|}{|\mathcal{I}^{\text{out}} \cup \mathcal{I}^{\text{tgt}}|} \quad (9)$$

where an instrument is considered present if there exists at least one note event using it.

Finally, **Voice Error Rate (VER)** evaluates the similarity of voice arrangements between two multitrack compositions. For each piece, we derive an ordered voice sequence by:

- 1) Computing the average pitch $\bar{p}_i$ for each active instrument $i \in \mathcal{I}^{active}$:

$$\bar{p}_i = \frac{1}{|T_i|} \sum_{e \in T_i} p_e \quad (10)$$

- 2) Constructing a voice sequence V by sorting instruments by descending average pitch:

$$V = [i_1, i_2, \dots, i_n] \text{ where } \bar{p}_{i_k} \geq \bar{p}_{i_{k+1}} \text{ for } k = 1, \dots, n-1 \quad (11)$$

The VER between output sequence V^{out} and target sequence V^{tgt} is:

$$VER = \frac{S + D + I}{N} \quad (12)$$

where S , D , and I are the minimum number of substitutions, deletions, and insertions required to transform V^{out} into V^{tgt} , and $N = |V^{tgt}|$ is the length of the V^{tgt} .

Among all the selected objective metrics, the $Note_i$ F1 is the most comprehensive and important one. To achieve a perfect $Note_i$ F1 score, the model need to perform perfectly in all evaluated aspects: instruments, voice, melody, note reconstruction and allocation.

C.2 Tokenization Evaluation

Average tokens per bar ($\bar{T}_{bar}$) and **average tokens per note** ($\bar{T}_{note}$) are computed by first counting the total number of tokens and the number of bars or notes, respectively, across the entire dataset. Specifically, $\bar{T}_{bar}$ is obtained by dividing the total number of tokens by the number of bars, while $\bar{T}_{note}$ is obtained by dividing the total number of tokens by the number of notes.

Bar-level Shannon Entropy ($\bar{H}_{bar}$) is computed by measuring the Shannon entropy of each bar-level token sequence $H(X)$:

$$H(X) = - \sum_{i=1}^N P(x_i) \log_2 P(x_i), \quad (13)$$

where X is the token distribution within a bar. Then we average it across all bars in the dataset to obtain $\bar{H}_{bar}$.

Note-level Perplexity (PPL_{note}) aggregates probabilities over all tokens representing a musical note (e.g., instrument, pitch, position, duration), and normalizes by the number of notes:

$$PPL_{note} = \exp \left(-\frac{1}{M} \sum_{j=1}^M \log P(n_j | n_{1:j-1}) \right), \quad (14)$$

where M is the number of notes in a bar.

Finally, **token-level Perplexity** (PPL_{token}) measures the standard autoregressive perplexity over all tokens. While commonly reported in language modeling, it serves as an auxiliary metric here, as it does not directly reflect the model's ability to model note events, which are basic units of music.

D Subjective Evaluation

D.1 Subjective Metrics

These metrics are based on listeners' auditory experiences and their subjective feelings while listening to the music. Since they are subjective, they cannot be easily defined using mathematical equations. However, we detail the evaluation criteria through the prompt questions provided in the questionnaire, which participants answered after listening to the demos. Each metric is assessed on a 5-point scale, ranging from 1 (very low) to 5 (very high).

- **Coherence:** Does the arrangement flow naturally and smoothly? How consistent is each instrument's performance and style throughout the piece?

- **Creativity:** How creative is the arrangement while maintaining faithfulness and naturalness?
- **Musicality:** What is the overall musical quality?
- **Faithfulness (band):** How closely does the arrangement resemble the original piece in terms of melody and overall feel?
- **Faithfulness (piano):** How closely does the arrangement capture the overall feel of the original piece?
- **Instrumentation (band-only):** Does each instrument fulfill its appropriate role within the band, and do they harmonize effectively?
- **Playability (piano-only):** How well is the piece suited for piano? How likely is it that a human pianist could perform this accompaniment?
- **Compatibility (drum-only):** Is the drum beat compatible with the other instruments?
- **Phrase Transition (drum-only):** How effectively does the drum arrangement handle transitions between phrases?

D.2 Human Evaluation Details

Survey Design. We conducted subjective evaluations using audio clips arranged by different models across three tasks: band arrangement, piano reduction, and drum arrangement. For band arrangement, we used out-of-domain test songs with novel compositions and instrument groups; for the other two tasks, test set songs were used to allow comparison with the original piano (Rule-O) or drum (ground truth) tracks. Each model was tasked with arranging the full song, and we selected a 15–30 second chorus phrase—the most representative segment—for evaluation. In total, we prepared 6 songs for band arrangement, 5 for piano reduction, and 5 for drum arrangement.

Each sample group includes the original music to be arranged and the anonymized outputs from all compared models (plus the ground truth for the drum task). Each sample is 8–16 bars long, rendered to audio in Cubase AI 13 using the default soundfont and the original MIDI’s BPM. We collected 56, 73, and 77 evaluation groups for band, piano, and drum tasks respectively. The mean time spent per session was approximately 30 minutes. Figure 6 shows the sample survey interface and instructions.

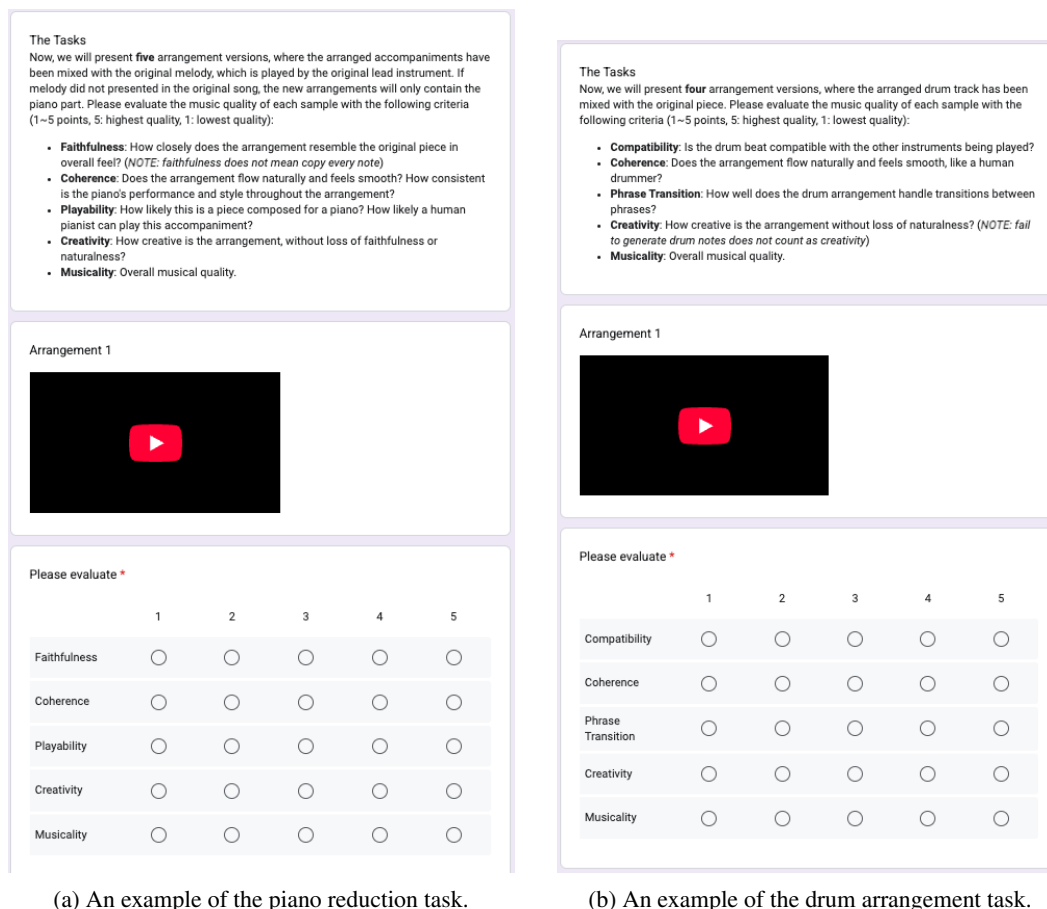
Participant Background A total of 26 participants joined the study, 5 of whom work in the music industry. Among all evaluators, 73.1% have over 10 years of experience in music composition or performance. We conducted t-tests comparing these raters with those having ≤ 10 years of experience (using piano reduction ratings on our model), and found no significant differences in the means between the two groups ($p > 0.2$ across all metrics). This suggests that musical background did not systematically bias the results.

Evaluation Protocol. Participants first listened to the original music clip, followed by model outputs presented in random order. For each arranged clip, participants rated multiple aspects using a 5-point Likert scale, based on task-specific questions in the previous subsection. The evaluation groups were randomly distributed to participants to ensure unbiased feedback. Each rater evaluated a subset of groups, where one group = one song (input) with multiple model outputs. On average, raters evaluated 2.15 groups for band, 2.80 for piano, and 2.96 for drum — corresponding to 8.6, 14.0, and 11.8 samples respectively.

Band Arrangement Settings. To evaluate generalization, we designed three distinct instrumentation settings: (1) string trio (violin, viola, cello), (2) rock band (synth lead, clean electric guitar, distorted electric guitar, electric bass), and (3) jazz band (saxophone, violin, brass section, clean electric guitar, piano, string ensemble, electric bass). Each setting was applied to two different songs, covering six songs in total. The input content was drawn from both piano arrangements (3 songs) and band arrangements (3 songs), testing the model’s ability to adapt to varying content and target instrument groups.

D.3 Discussion on Evaluation Scheme

Our experiments primarily adopt a 5-point scale for subjective evaluation, which is the most commonly used protocol in prior music arrangement works [42, 35, 38]. However, we observe a growing trend toward using A/B testing to evaluate overall quality in the broader music generation literature [32,



(a) An example of the piano reduction task.

(b) An example of the drum arrangement task.

Figure 6: Screenshots of survey pages and instructions of our online survey.

23, 37]. We encourage future research to consider combining both approaches to obtain more comprehensive and convincing evaluation results.

E Probing Analysis of Pre-Training Impact

Knowledge probing techniques are used to explore what a model has learned within its hidden representations [27]. A widely adopted method is linear probing, where the pre-trained model is frozen and a simple linear classifier is trained on top of the hidden representations to predict specific properties, thereby revealing the extent to which particular knowledge is encoded in the model [1]. This technique is particularly useful in our context for evaluating what the model has internalized during pre-training.

We have shown in the paper that the proposed models outperform the baseline models that do not undergo the pre-training stage on the generation quality. In this section, we further analyze the reason. Specifically, we conduct probing experiments to assess whether pre-training enhances the acquisition of musical knowledge to facilitate understanding content conditions in fine-tuning.

We focus on two probing tasks: (1) classifying instrument types from content sequences that contain only position, pitch, and duration tokens, and (2) recognizing chord progression sequences from content sequences. To determine whether instrument or chord information is linearly accessible within the model's sequence embeddings, we employ linear classifier probes. In these probing experiments, the average pooling of the Transformer's output embeddings across all tokens in a sequence is fed into the classifier. The model parameters are kept frozen, and only the linear classifiers are trained. We compare the knowledge captured by different models: a randomly initialized Transformer, a model that has undergone pre-training only (PT only), a model trained on the band arrangement task

Table 8: Instrument probing results.

Model	Acc@1	Acc@3	Acc@5
Random guess	2.94	8.82	14.71
Random initialized	38.98	61.71	74.22
FT only	41.50	64.93	76.53
PT only	46.14	69.30	79.61
PT + FT	45.89	68.96	79.47

Table 9: Chord probing results.

Model	Chord Root		Chord quality	
	Acc@1	Acc@3	Acc@1	Acc@3
Random guess	8.33	25.00	11.11	33.33
Random initialized	48.86	78.76	38.05	77.97
FT only	50.09	80.23	39.26	79.03
PT only	62.93	89.42	50.09	85.58
PT + FT	58.05	85.93	44.92	82.48

without pre-training (FT only), and a model that has undergone both pre-training and fine-tuning (PT + FT).

For the probing experiments, the batch size was set to 12 for chord probing and 64 for instrument probing. The learning rates were $5e-4$ for chord probing and $1e-4$ for instrument probing. Both experiments shared the same remaining hyperparameters: training for 10 epochs, using a linear learning rate scheduler, a 500-step warmup, and a weight decay of 0.01.

E.1 Instrument Type Probing

In this task, we use a linear probe to estimate the instrument type from a single-track music sequence without providing instrument tokens. The goal is to predict which instrument is most likely to play the given note sequence. The performance is evaluated using top-1, top-3, and top-5 prediction accuracy metrics.

As shown in Table 8, a model initialized with random weights shows notable improvement after fine-tuning, suggesting that the ability to discern instrument styles is the requirement for performing well in music arrangement tasks. However, the gains in accuracy are modest, likely due to the limited number of training samples, which may constrain the model’s ability to acquire such knowledge through fine-tuning alone. Interestingly, models that underwent pre-training exhibit the highest accuracy in predicting instrument types. This indicates that substantial knowledge of instrument styles can be acquired effectively during the pre-training phase. Moreover, the proposed models that are both pre-trained and fine-tuned (PT+FT) maintained high accuracy levels, demonstrating that the knowledge about instruments styles are useful for the arrangement task.

E.2 Chord Progression Probing

In this task, we use linear probes to predict chord progressions from a 2-bar music content sequence without instrument tokens. Eight linear probes are trained simultaneously to predict the chord roots and qualities for a total of four chords (two chords per bar). This task is used to evaluate whether the model contains position-specific chord information. Similarly, we use top-1 and top-3 prediction accuracy metrics.

As shown in Table 9, the model with only fine-tuning (FT only) indicates a foundational grasp of chord knowledge for this complex music analysis tasks. However, similar to instrument prediction, the knowledge of recognizing chord progression does not significant gain, until the pre-training is also introduced into the model, confirming that pre-training establishes a robust basis for better understanding of content sequence, which may potentially help with the quality of arrangement.

F Broader Impact

This work presents a unified framework for symbolic music arrangement using pretrained generative models, achieved by a reconstruction fine-tuning objective and a structured tokenization scheme. By enabling high-quality arrangement generation under flexible control, our approach has the potential to make music creation more accessible to non-experts, reduce the technical burden for composers, and support educational and assistive tools for learning music theory, orchestration, or instrumentation. It may also benefit creative professionals by streamlining workflows in game audio, film scoring, and digital content production.

Many other conditional generation tasks could similarly benefit from this training objective design. For instance, one can construct a conditional sequence derived from a partial or aspect-wise decomposition of the target, concatenate it with the original target tokens, and then perform segment-level generation using a pretrained model with historical context-awareness to fine-tune existing generative models. Examples include bar-level infilling (e.g., removing and rewriting one bar), melody generation conditioned on chords, variation generation from a given melody, counter-melody generation, or harmonizing a melody with chords. All of these tasks share the same music-conditioned generation pattern and may benefit from a strong generative pretrained model.

Beyond arrangement tasks, the proposed tokenization scheme may have broader implications for symbolic music modeling in general. By restructuring musical data into continuous track sequences, it facilitates learning over reusable musical phrases. This may improve both conditional and unconditional generation quality across tasks such as continuation, accompaniment generation, and reharmonization. Moreover, we anticipate that structured representations like REMI-z could benefit music understanding tasks, including symbolic transcription from audio, by facilitating the modeling of instrument-wise playing styles during decoding. Additionally, one may consider applying sequence compression techniques—such as Byte Pair Encoding (BPE) or variational autoencoders (VAE)—on top of this base tokenization to enhance compactness, which could further support long-range modeling essential for full-song generation. We encourage future work to explore these directions.

Beyond arrangement tasks, the proposed tokenization scheme may have broader implications for symbolic music modeling in general. By restructuring musical data into locally coherent, instrument-consistent sequences, it facilitates learning over meaningful musical units. This could improve both conditional and unconditional generation quality across tasks such as continuation, accompaniment generation, or reharmonization. Moreover, we anticipate that structured representations like REMI-z may benefit music understanding tasks, including symbolic transcription from audio, by facilitating modeling of instrument-wise playing styles when decoding. Also, you can consider combine sequence compression techniques, maybe BPE, maybe VAE, on top of that, to further enhance the compactness, which may facilitate long-range modeling that are important to full-song generation. We encourage future work to explore these directions.

However, there are potential risks. As with other generative models, the misuse of automatic arrangement systems could devalue human artistry if deployed without appropriate attribution or transparency. The system may also reflect and amplify stylistic biases present in the training data, which primarily consists of western popular music. This could marginalize underrepresented musical traditions or reinforce narrow definitions of musical aesthetics. We encourage future researchers and practitioners to consider ethical deployment strategies, such as transparent model labeling, dataset diversification, and collaborative workflows where AI augments rather than replaces human creativity.

G Limitations

Fine-tuning dataset focused on pop genres. While our framework is designed to be genre-agnostic, our fine-tuning and evaluation primarily focus on pop-style arrangements. Future work can further explore the generalizability across a wider range of genres such as classical, jazz, or non-Western traditions.

Strict structural alignment assumption. Our framework assumes that the input and output music share identical musical structure, which enables a segment-to-segment reconstruction formulation. While effective, this simplification does not fully reflect the broader musicological concept of

arrangement, which often involves modifying the musical form or phrase structure. This constraint is not unique to our method—most prior works also rely on such structural alignment—but relaxing this assumption remains an important direction for future research.

Lack of instrument planning. Although our system allows users to specify the instruments and their voice relationships for each segment during inference, the global planning of these assignments is not addressed. In practice, achieving a musically satisfying arrangement often requires deliberate instrument planning across segments—e.g., rotating lead instruments, or changing instrument combinations between sections. This work focuses on controllability at the segment level and leaves high-level planning strategies for future exploration.

Melody retention in piano reduction is not guaranteed. While our piano reduction method captures harmonies and textures from ensemble music, it often fails to retain the melody. This is largely because the original piano track in multitrack data typically does not carry the main melodic line, and our current training setup does not include any explicit melody supervision. Solving this may require incorporating human-created piano arrangements or explicitly modeling melodic salience, which we leave for future work.

Handling of Time Signature and Tempo. Our current model does not explicitly model time signature or tempo tokens during fine-tuning. As a result, it is limited to handling music with a fixed 4/4 time signature and stable tempo. This design choice follows common practice in prior arrangement research [42, 35, 30], where similar assumptions are made to simplify modeling. Empirically, approximately 95% of the fine-tuning data is in 4/4 time (see Appendix A), and stable tempo is generally the norm in modern music [7]. Nevertheless, extending the framework to handle non-4/4 time signatures and dynamic tempo variations remains an important direction for future work.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: We ensure the claims in abstract and introduction accurately reflect our contributions and supported by experiments.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: Limitations are summarized in Appendix G.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[NA\]](#)

Justification: Our paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: We introduced our experimental settings in § 4.1 and provide details of hyperparameter settings necessary for reproduction in Appendix B.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: All datasets used in this work are open-source. Please refer to the code in the supplementary material, which will be released upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We introduced our experimental settings in § 4.1 and provide details of hyperparameter settings necessary for reproduction in Appendix B.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We adopted within-subject (repeated-measures) ANOVA significance test for subjective evaluation, and presented mean and standard error in the tables. For objective evaluation, we used Wilcoxon signed rank test to support the major claim (tokenization advantage).

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The compute resources are stated in § 4.1 and detailed in Appendix B.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The authors have reviewed and conformed in every respect with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discuss both positive and negative impacts of our work in Appendix F.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: All datasets used in this paper are open-source datasets with minimum risk of misuse, and so are the models trained on them.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We use two datasets for model training: Los Angeles MIDI Dataset is licensed under Apache-2.0 License, and Slakh2100 is licensed under CC BY 4.0.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: The code in the supplementary material is well documented.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[Yes\]](#)

Justification: We provide crowdsourcing details including screenshots in Appendix D.2.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[Yes\]](#)

Justification: The human study in our experiment is based on online crowdsourcing, which bears minimum risk. Participants are informed that participation in our study is entirely voluntary and that they may choose to stop participating at any time without any negative consequences. No personally identifying information is collected in the human study.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We only used LLM for writing and formatting of the manuscript.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.