MuLaCover: Flexible Cover Song Generation via Symbolic Tonal Control

Lynn Yi[♯*] **Zihan Xiong**[×*] **Junjie Cao**[♭*] **Xiaolong Weng**[♭*] **Yingzhe Ma**[♭]
Gus Xia[♯] **Jinliang Liu**[♭] **Yuxin Xie**[♭] **Xianwei Zhuang**[♭] **Yuguo Yin**[♭]
Qiao Jin[♭] **Gongxi Zhu**[♭] **Jiayu Wang**[♭] **Junjie Liang**[×] **Qin Yue**[♮]
Ziyu Wang[♯] **Dading Chong**[♭] **Meng Cao**[♯] **Jiahuan Zhou**[×] **Dongchao Yang**[♭♭†]
[♯]MBZUAI [×]Peking University [♭]MuLa Labs, Vera Praxis Lab
[♭♭]The Chinese University of Hong Kong [♮]Nanjing University of the Arts

Abstract

A cover song re-renders an existing piece, most commonly by preserving its tonal content—melody and chord progression—while reshaping other musical attributes. This suggests a natural formulation for generative cover synthesis with pretrained music language models: explicitly control tonal content, while specifying lyrics and style through text. Existing approaches condition on frame-level features, tightly coupling generation to the reference's timing and structure, and limiting flexibility such as tempo variation, structural rearrangement, or partial conditioning. To this end, we present MuLaCover, a cover generation model that conditions a pretrained text-to-song foundation model on an editable symbolic lead sheet. Our design enables alignment-free control: generated outputs preserve the tonal signature of the source without frame-level alignment, allowing flexible timing, structure, and segment-level generation. We further introduce a training curriculum with partially mismatched audio–symbolic pairs, improving diversity and robustness. We evaluate MuLaCover with objective metrics, standard subjective ratings, and in-depth expert interviews that provide fine-grained diagnostic insights. Experiments show that MuLaCover achieves state-of-the-art performance among open-source systems and is competitive with leading commercial models such as Suno-v5.5. Audio samples and codes are available at our project page `https://mulalabs.ai/mulacover/`.

1 Introduction

A *cover song* is a re-rendering of an existing piece that preserves its musical identity while freely reshaping every other aspect of the recording. Decades of cover-song-identification research [1–7] converge on a clean characterization of this identity: across renditions, key, tempo, instrumentation, vocal timbre, song structure, and even lyrics may all change; what typically stays invariant is the *tonal content* — the melody and the underlying chord progression. This invariant/variant taxonomy suggests a natural recipe for cover *generation*: condition a pretrained text-to-song foundation model [8–10] on the tonal content of a source rendition, and let its lyrics and style prompts shape everything else.

The unresolved design question is how to encode the tonal-content signal. Existing audio-domain cover systems answer it with frame-level features extracted from the source recording itself, such as vocal F_0 contours [11] or quantized audio latents [12], sampled at the same rate as the target audio. This choice imposes a strict *frame-wise alignment* between source and cover: the cover cannot run at

*Equal contribution. `xiaoyu.yi@mbzuai.ac.ae`
†Corresponding author. `dcyang@se.cuhk.edu.hk`

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

a different tempo, cannot rearrange structure, and cannot draw on a partial reference such as a single verse or chorus.

We argue that the right tonal-content representation is the *symbolic lead sheet* long used in music information retrieval [13–15]: a melody line paired with a chord progression, which by construction abstracts away surface acoustics and retains only the typically cover-invariant musical skeleton. Crucially, before feeding the lead sheet into the model, we time-stretch it so that all sequences share a canonical tempo. This BPM normalization is what frees the conditioning signal from any frame-wise tie to the source recording: it leaves the cover model free to set its own pace, and, via attention masking over the symbolic sequence, lets the model accept any sub-segment of the song as a partial reference. We inject this normalized symbolic signal into a pretrained text-to-song backbone through a gated cross-attention module, yielding our system **MuLaCover**.

Two issues remain after the architectural design. The first concerns data. Most text-to-song models [8, 16, 17, 9–11] are trained on recordings whose lyrics, style, and audio necessarily come from the same single rendition; the model has never been asked to render the tonal content of one rendition under the lyrics and style of a *different* one, yet precisely this mismatch is what cover generation demands. We close this supervision gap through *cross-cover finetuning* on a curated paired dataset that pairs the tonal content of one rendition with the lyrics, style, and audio of another. The second concerns evaluation. Cover quality is multi-faceted and idiomatic, but an aggregate Mean Opinion Score (MOS) listening study reveals *whether* a system is good without explaining *why*, and offers little guidance for system iteration. We therefore complement listening-study questionnaires with a structured *expert-interview* protocol that elicits diagnostic, comparative judgments from trained listeners. Together with objective metrics, these subjective signals yield a comprehensive evaluation.

We evaluate MuLaCover on two cover tasks. On the standard *full-song cover* task, MuLaCover is state-of-the-art among open-source systems on objective metrics of musical quality, lyric and style controllability, and cover identifiability, and is broadly on par with the leading closed-source commercial system Suno-v5.5; the listening study and expert interviews corroborate these findings. As a natural consequence of its alignment-free conditioning, MuLaCover further tackles *partial-reference cover generation*, in which only a fragment of the reference is provided and the model produces a full-song cover that preserves the fragment's tonal content while freely re-creating the rest, a task on which MuLaCover achieves superior performance over existing open-sourced baselines across three fragment configurations in objective evaluation.

Our main contributions are:

- We formulate cover generation as conditioning a pretrained text-to-song foundation model on the BPM-normalized symbolic lead sheet of a source rendition, and realize it as MuLaCover, which injects the signal through a gated cross-attention module. The design removes the strict frame-wise alignment that has bound prior cover systems to the source's exact timing and structure, and supports partial-reference cover generation as a natural by-product.
- We identify cross-version pairing of content and style as the missing supervision signal for cover generation, and provide a curated paired dataset together with a cross-cover finetuning curriculum that turns a generic text-to-song model into a cover generator.
- We evaluate MuLaCover with objective metrics, listening-study questionnaires, and a structured expert-interview protocol that complements aggregate scores with fine-grained, comparative diagnostics. The combined evaluation establishes MuLaCover as state-of-the-art among open-source cover systems and broadly on par with the closed-source commercial system Suno-v5.5.

2 Related Work

2.1 Cover Song Identification

Cover song identification (CSI) is the music information retrieval task of retrieving cover versions of a query song from a database [1]. Across versions, key, tempo, structure, instrumentation, timbre, and lyrics may all change [1–7]; the most stable cross-version identifier is the *tonal content* [3], namely the melody [18] and the underlying chord progression or chroma features [19, 2], which serves as

the primary cross-version identifier. This invariant-vs-variant decomposition has shaped algorithmic design throughout the CSI literature. Classical rule-based systems [2, 3] extract tonality-oriented descriptors such as chroma features [20] or HPCP sequences [21], and pair them with similarity computations that explicitly account for the varying attributes: transposition handles key shifts, and dynamic time warping handles tempo differences [3]. Modern deep CSI systems [4, 22, 5–7] preserve the same inductive bias but realise it through learned representations, training networks on contrastive or classification objectives with pooling, matching, and augmentation strategies (e.g. frequency-axis rolling for pitch shifts, speed perturbation, and spectrogram masking) deliberately designed so that embeddings remain invariant to key, tempo, and structural variations between versions. Viewed through a generative lens, the variant/invariant taxonomy of CSI is exactly a *content/style* decomposition of a song: tonal content is the content, and everything else (instrumentation, timbre, tempo, structure, lyrics) is the style that a cover artist typically reshapes, although where this line is drawn varies from cover to cover. Cover song generation is then the *style transfer* counterpart of CSI, and as such belongs to the broader space of controllable music generation reviewed next.

2.2 Controllable Music Generation

In the audio domain, controllable music generation has advanced rapidly through language-model and diffusion systems [23, 24, 16, 9, 25, 8, 17, 10]. Control here is predominantly textual—style descriptions and lyric prompts—with finer-grained content signals such as melody and chord progression only recently emerging as auxiliary conditions [26–28, 11]. Symbolic music generation, by contrast, has long offered interpretable, disentangled control at global (time signature, tempo, instrumentation [29–31]), local (chord progression, compositional texture [32–34]), and hierarchical [35] scales. For cover song generation, symbolic piano cover systems [36–38] remain narrow in scope. Audio-domain systems such as SongEcho [11] and ACE-Step 1.5 [12] condition on a temporal feature of the source sampled at the same rate as the target audio features, enforcing strict frame-wise alignment and leaving no degree of freedom in tempo or structure. MuLaCover removes this rigidity through flexible symbolic lead-sheet conditioning.

Since music composition is inherently a cascaded process, recent works chain multiple specialised modules, each handling one sub-task (e.g. melody writing, harmonisation, structural planning, singing synthesis, or sample retrieval) and passing intermediate symbolic or structural representations to the next [39, 40]. While such pipelines expose interpretable and editable intermediate stages, errors accumulate across modules and the final audio is bounded by the individual components that produce it. As a lead-sheet-conditioned audio generator, MuLaCover can be intergrated naturally into such pipelines to better support human–AI music co-creation.

3 Method

MuLaCover equips a decoder-only text-to-song backbone with tonal-content control. Mainstream text-to-song foundation models [8, 10] already expose two controls — a lyrics prompt ℓ along with a style prompt s that conveys non-lyric attributes such as genre and instrumentation — and autoregressively model the audio token sequence a as their generation target. MuLaCover adds the tonal-content signal c on top of these existing controls; an overview of the resulting architecture is shown in Figure 1(a).

Abstracting away the specific text-to-song instantiation, the remaining design problem becomes one of sequence-to-sequence conditional injection. The backbone produces, at each of its N transformer layers, a sequence of hidden states $h_l \in \mathbb{R}^{S\times D}$ ($l = 1, \ldots, N$, with S the sequence length and D the feature dimension); the new control is itself a sequence $c \in \mathbb{R}^{T\times D}$ of length T. MuLaCover merges information from c into h_l at the last L layers via a gated adaptive cross-attention module, leaving the remaining $N-L$ layers and the original training loss intact.

Three components, each ablated independently in Section 4.5, complete the system: (i) a symbolic lead-sheet representation that defines the tonal-content sequence c (Section 3.1); (ii) the gated adaptive cross-attention injection module (Section 3.2); and (iii) a three-stage curriculum that aligns the newly introduced parameters with the backbone and exposes the model to cross-version training pairs in which the tonal-content signal and the existing controls are drawn from different renditions of the same song (Section 3.3).

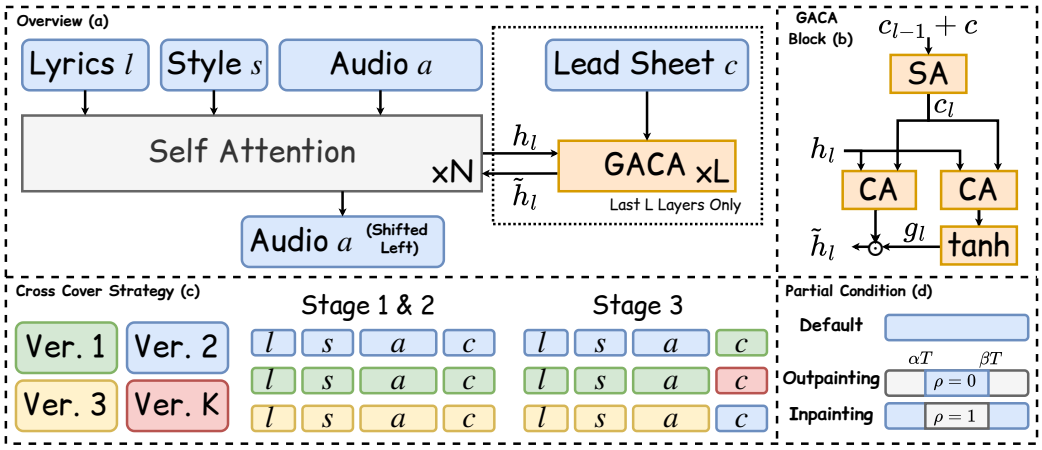


Figure 1: Overview of MuLaCover. **(a)** The pretrained text-to-song backbone has N transformer layers consuming lyrics ℓ, style s, and audio tokens a; at its last L layers, the Gated Adaptive Cross-Attention (GACA) module injects the symbolic tonal-content condition c into the hidden states h_l. **(b)** Inside a GACA block, c is refined by a layer-specific self-attention (SA) into c_l, then merged into h_l through a cross-attention (CA) update modulated by a learned, $\tanh$-bounded gate g_l. **(c)** Cross-cover strategy in Stage 3: colors denote different renditions of the same song. Stages 1–2 draw all four signals ℓ, s, a, c from the same rendition; Stage 3 draws c from a different rendition than ℓ, s, a. **(d)** Partial conditioning via attention masking on c: ρ=0 keeps the interval $[\alpha T, \beta T]$ (*outpainting*), ρ=1 keeps its complement (*inpainting*); the default (α, β, ρ)=$(0, 1, 0)$ recovers the full-song setting.

3.1 Symbolic Lead Sheet Representation

A lead sheet, comprising a melody line and a chord progression, captures the tonal content essential to the cover task and has been used in music generation [13, 37, 39]; we adopt it as our conditioning signal c.

Given a reference recording, we extract the vocal melody and chord progression with pretrained transcription models [41, 42]. To prevent the lead sheet from inheriting the source's specific tempo, and thereby to avoid imposing any frame-wise alignment between c and the generated audio, we estimate the source's beats-per-minute (BPM) and time-stretch the lead sheet so that all sequences are normalized to a canonical BPM=120. We further quantize the symbolic timeline at the resolution of a sixteenth note and denote the resulting sequence length as T. This BPM normalization is essential to the system: removing it forces a strict frame-wise correspondence between c and the audio that exposes a sharp content–style tradeoff, as we quantify in Section 4.5 (w/o BPM-norm).

After BPM normalization, the melody is encoded as a binary pianoroll $M \in \{0,1\}^{T\times 256}$, whose $256 = 128 \times 2$ feature dimensions capture the onset and sustain status of each of the 128 MIDI pitches at every step; the chord progression is encoded as a binary chromaroll $H \in \{0,1\}^{T\times 12}$ over the twelve pitch classes. The two representations are projected into a shared embedding space and concatenated along the feature dimension to form the conditioning sequence consumed downstream:

$$c \;=\; \big[\, M\,W_M \,\|\, H\,W_H \,\big] \;\in\; \mathbb{R}^{T\times D}, \tag{1}$$

where $W_M \in \mathbb{R}^{256\times D/2}$ and $W_H \in \mathbb{R}^{12\times D/2}$ are learned linear projections, $\|$ denotes feature-wise concatenation, and D matches the feature dimension of the backbone. Conditioning on the melody alone, without the chord progression, is ablated in Section 4.5 (w/o Chord); we also compare the symbolic lead sheet against the frame-wise F_0 contour used by prior cover systems [11] in the same section (w/ F_0).

3.2 Gated Adaptive Cross-Attention Injection

MuLaCover injects the symbolic condition c only into the last L layers of the backbone — layers $N-L+1$ through N. The intuition is that adding tonal-content control should only need to reshape the predicted audio-token distribution at the final stages of generation; the lower $N-L$ layers, which carry the pretrained backbone's general acoustic representations, can therefore be left structurally unchanged. The internal structure of the resulting GACA block is illustrated in Figure 1(b). We compare this cross-attention-based design against the alternative prefix-tuning mechanism used in prior LM-based controllable music generation work [26, 28] in Section 4.5 (w/ Prefix tuning).

At each modified layer l, the symbolic condition is first refined by a layer-specific *self-attention* block, denoted $\mathrm{SA}_l(\cdot)$, which allows the conditioning to specialize to the level of abstraction expected by the corresponding decoder layer. The previous refined condition at layer $l-1$ is also fed into it to improve the expressivity:

$$c_l \;=\; \mathrm{SA}_l(c + c_{l-1}) \;\in\; \mathbb{R}^{T\times D}. \tag{2}$$

Note that c_{l-1} is omitted at the first injecting layer $l = N - L + 1$. Here and throughout, $\mathrm{SA}(\cdot)$ stands for a standard multi-head self-attention block [43].

The refined symbolic states c_l are then merged into h_l through a *cross-attention* block, which we denote $\mathrm{CA}_l(q, m)$: q is the query input and m is the source input, from which the keys and the values are derived; the output has the same length as q. Before c_l is fed in as the source input, we apply rotary positional embeddings (RoPE) [44] to it, so that the temporal positions of the symbolic sequence are carried through into the keys and values together with their content. The query input is the audio-side hidden states h_l.

A naive residual update of the form $\tilde{h}_l = h_l + \mathrm{CA}_l(h_l, c_l)$ admits the conditioning update at every position and feature dimension indiscriminately, leaving no room for the model to decide where the symbolic information should actually be incorporated; we observe in Section 4.5 (w/o Gate) that this substantially degrades cover identifiability. Inspired by [26], we therefore introduce a *learned gate* $g_l \in (-1, 1)^{S\times D}$ that lets the model decide adaptively, for each position and feature dimension, how much symbolic information to admit at layer l:

$$\tilde{h}_l \;=\; h_l \;+\; g_l \;\odot\; \mathrm{CA}_l\big(h_l, c_l\big), \tag{3}$$

$$g_l \;=\; \tanh\big(\mathrm{CA}_l^{\mathrm{gate}}\big(h_l, c_l\big)\big), \tag{4}$$

where $\odot$ denotes elementwise multiplication, $\tanh$ bounds the gate to $(-1, 1)$, and $\mathrm{CA}_l^{\mathrm{gate}}$ is a parameter-light cross-attention block that shares the same (q, m) convention as CA_l and likewise receives the RoPE-augmented c_l as its source input. We zero-initialize the output projection of $\mathrm{CA}_l^{\mathrm{gate}}$, so that $g_l = \tanh(0) = 0$ at the start of training and the augmented model exactly reproduces the pretrained backbone's behavior at initialization.

Partial conditioning via attention masking. The condition c is built from the full-song lead sheet, yet MuLaCover also supports covers of arbitrary song segments through attention masking, naturally instantiating the two canonical partial-generation modes. The user specifies a mask interval via two scalars $\alpha, \beta \in [0, 1]$ with $\alpha \le \beta$, together with a binary inversion flag $\rho \in \{0, 1\}$: ρ=0 keeps the interval as the only conditioned region (*outpainting*, in which the model is asked to freely generate the rest of the song around a given segment), while ρ=1 keeps its complement (*inpainting*, in which the model fills in the masked interval given the surrounding conditioning context). These parameters induce a set of *active positions*

$$\mathcal{A}(\alpha, \beta, \rho) = \begin{cases} \{\, t \in \{1, \dots, T\} : \alpha T < t \le \beta T \,\}, & \rho = 0 \quad \text{(outpainting)}, \\ \{1, \dots, T\} \setminus \{\, t \in \{1, \dots, T\} : \alpha T < t \le \beta T \,\}, & \rho = 1 \quad \text{(inpainting)}. \end{cases} \tag{5}$$

Within SA_l, self-attention is restricted to position pairs both lying in $\mathcal{A}$; within CA_l and $\mathrm{CA}_l^{\mathrm{gate}}$, queries from h_l may only attend to keys and values whose source position lies in $\mathcal{A}$. Positions outside $\mathcal{A}$ are thus ignored entirely by the injection pathway. The defaults $(\alpha, \beta, \rho) = (0, 1, 0)$ recover the full-song setting; the three masking modes are illustrated in Figure 1(d).

3.3 Curriculum Training Strategy

We train MuLaCover in three stages. Stages 1 and 2 establish the mapping from the lead sheet to the audio at increasing scale; Stage 3 exposes the model to cross-version pairings of content and style.

Stage 1: Short-clip warmup. We first train on 30-second segments. Short clips are cheap to optimize and let the newly introduced modules pick up the basic mapping from the lead sheet to the audio quickly, before the model is asked to handle longer-range structure. Skipping this stage degrades both perceptual quality and cover identifiability, as we show in Section 4.5 (w/o Warmup).

Stage 2: Full-song training. We then continue training on full-length recordings, extending the mapping learned in Stage 1 to song-scale temporal context. The existing controls ℓ, s, the tonal-content signal c, and the audio target are all drawn from the same song in Stage 1 and 2.

Stage 3: Cross-cover finetuning. A cover song takes the tonal content of one rendition and re-renders it under the lyrics and style of another. Stages 1 and 2 do not expose the model to this cross-version pairing, since c and ℓ, s are always drawn from the same recording. To close this gap, we collect, for each song, a group of K available renditions $\{v_1, \ldots, v_K\}$, and at each training step sample two indices $i, j \in \{1, \ldots, K\}$: the tonal-content signal c is extracted from rendition v_i, while the remaining controls ℓ, s and the audio target are taken from rendition v_j. The model is thus explicitly trained to render the tonal content of v_i under the lyrics and style of v_j, which is precisely the cover-generation behavior we wish to evoke at inference. Figure 1(c) contrasts this Stage 3 cross-version pairing with the same-rendition setup of Stages 1 and 2. The contribution of this cross-cover pairing to style controllability is quantified in Section 4.5 (w/ SP).

4 Experiments

This section describes the training setup of MuLaCover and conducts evaluation on the full-song cover task and partial-reference cover task. Through objective and subjective evaluation, we show that MuLaCover surpasses representative open-source systems and is on par with the strongest closed-source model. We conduct ablation studies to show the necessity of each design choice.

4.1 Training Configurations

Following the data-construction protocol of recent work [11, 45], we obtained approximately 200,000 synthetic songs from music generation providers. Each track is passed through an automated annotation pipeline: vocals are isolated with Demucs [46] and transcribed with Whisper [47]; section-level structure is obtained by ensembling All-in-One [48] and SongFormer [49]; and free-form style tags produced by Qwen2-Audio [50] are formatted with DeepSeek [51] into the tag vocabulary used by HeartMuLa [10]. Lead sheet is transcribed using YourMT3 [41] for melody track and [42] for chord progression. For Stage 3, we additionally curate a paired cross-cover set: for each of 1,250 source songs we query music generation providers to synthesize 7 stylistically distinct covers, yielding 1,250 clusters of 8 interpretations each (the source plus 7 covers). The full data construction pipeline for Stage 3 is described in Appendix A.

We build MuLaCover on the 3B-parameter variant of HeartMuLa [10], an LM-based music generation model with $N = 28$ backbone transformer layers. Our Gated Adaptive Cross-Attention modules are inserted into the last $L = 8$ backbone layers. Stages 1 and 2 keep the backbone frozen and update only the newly introduced parameters; Stage 3 additionally applies LoRA [52] to the last $L = 8$ backbone layers and finetunes on the cross-cover pairs. All experiments run on 16×NVIDIA A100 (80 GB) GPUs; per-stage hyperparameters are listed in Appendix B.

4.2 Tasks and Baselines

We evaluate MuLaCover on two cover-generation tasks. Task 1 is full-song cover generation, where the model is asked to generate a cover conditioned on the reference music, target style, and corresponding lyrics. Task 2 is partial-reference cover generation. Under this setting, only a fragment of the reference music is observed, along with the target style and lyrics for the whole song. The model is asked to produce a full-song cover, preserving the tonal content of the observed part while recreating the remainder. Instantiating the active-position mask $\mathcal{A}(\alpha, \beta, \rho)$ from Eq. 5, Task 2 admits three setups—continuation, outpainting, and inpainting—detailed in Appendix C.

We compare against three contemporary systems: SongEcho [11], a fine-tuning-based cover system; ACE-Step-1.5 [12], a large-scale pretrained music foundation model; and Suno-v5.5 [53], a leading closed-source commercial system. See Appendix C for baseline configurations. Both Task 1 and

Table 1: Objective evaluation on the full-song cover task (Task 1). ↑ / ↓ indicate higher / lower is better. **Bold** marks the best and underline the second-best result in each column.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
SongEcho [11]	7.88	6.11	7.26	7.54	3.70	3.57	3.37	3.47	3.46	0.26	0.38	0.39	0.88	1.55
ACE-Step-1.5 [12]	8.29	6.30	7.68	7.86	4.48	4.46	4.26	4.35	4.36	0.21	0.30	0.40	0.95	1.10
Suno-v5.5 [53]	8.30	**6.44**	7.66	7.92	**4.61**	**4.60**	**4.42**	**4.52**	**4.51**	**0.11**	**0.50**	0.33	**0.98**	1.11
MuLaCover	**8.34**	6.34	**7.70**	**7.93**	4.59	4.59	4.34	4.44	4.47	0.13	0.41	**0.48**	0.97	**1.05**

Table 2: Objective evaluation on the partial-reference cover task (Task 2), averaged over the three masking configurations (continuation, inpainting, outpainting). ↑ / ↓ indicate higher / lower is better. **Bold** marks the best.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
SongEcho [11]	7.58	5.59	6.80	7.44	3.14	3.04	2.85	2.87	2.92	0.55	0.25	**0.63**	0.41	5.94
ACE-Step-1.5 [12]	8.26	5.91	7.66	7.84	4.22	4.20	3.95	4.08	4.08	0.36	0.34	0.50	0.52	4.92
MuLaCover	**8.33**	**6.31**	**7.69**	**7.92**	**4.57**	**4.57**	**4.32**	**4.42**	**4.45**	**0.12**	**0.42**	0.50	**0.70**	**2.60**

Task 2 are evaluated objectively in Section 4.3; for Task 2, objective metrics are averaged across the three setups, with per-setup metrics reported in Appendix C. Task 1 additionally receives a subjective evaluation in Section 4.4. Suno-v5.5 does not support the partial-reference setting and is therefore omitted from comparison in Task 2.

4.3 Objective Evaluation

We prompt DeepSeek [51] to generate 32 sets of lyrics and feed them to Suno-v5.5 [53] to obtain 32 reference songs, and prompt DeepSeek again to generate 8 distinct style tags. Each model under evaluation is conditioned on a reference song, its lyrics, and one style tag at a time, producing 8 stylistic variants per reference song for a total of 256 covers per model.

We assess perceptual quality with two pretrained predictors. From AudioBox-Aesthetics [54] we report Production Quality (PQ), Production Complexity (PC), Content Enjoyment (CE), and Content Usefulness (CU). From SongEval [55] we report Coherence (Coh.), Memorability (Mem.), Naturalness (Nat.), Clarity (Cla.), and Musicality (Mus.). For lyric controllability, we isolate the vocal track of each cover with Demucs [46], transcribe it with Whisper [47], and report Phoneme Error Rate (PER) against the input lyrics. For style, we use the MuQ-MuLan music–text contrastive model [56]: MuQ-Sim is the cosine similarity between the cover and its target style tag in the joint embedding space, measuring fidelity to the requested style; MuQ-Dist is the cosine distance between the cover and the source recording, measuring stylistic deviation from the source. For cover identifiability, we utilize cover song identification model CoverHunter [7]: each cover serves as a query against the database of 32 source songs, and we report top-1 retrieval accuracy (CSI-R@1) and the mean retrieval rank of the true source (CSI-Rank).

Table 1 reports objective results on Task 1. Overall, MuLaCover outperforms the open-source baselines on most metrics and performs on par with the closed-source commercial system Suno-v5.5. For audio quality, MuLaCover generates songs whose production quality and musicality are comparable to those of the commercial systems and generally better than those of the open-source baselines, suggesting that adding fine-grained controls does not compromise generation quality. For controllability, MuLaCover follows the given lyrics much more faithfully than the open-source baselines and nearly matches Suno-v5.5. In terms of style, it departs from the source arrangement more substantially than all baselines while maintaining reasonable fidelity to the target style, and the CSI results confirm that the generated songs remain recognizable as covers of the source song. In other words, MuLaCover changes the style without losing the identity of the original song.

Table 2 reports objective results on Task 2 averaged over the three masking configurations (continuation, inpainting, outpainting); per-configuration breakdowns are deferred to Appendix D. When only a fragment of the reference music is exposed, the baselines suffer a sharp drop in cover identifiability, accompanied by varying degrees of degradation in lyric controllability and audio quality. MuLaCover, in contrast, degrades far more gracefully and achieves the best performance on all metrics except MuQ-Dist, where SongEcho's higher value coincides with its near-total loss of identifiability

Table 3: Listening study: mean $\pm$ 95% confidence interval on each metric (5-point Likert; higher is better). **Bold** marks the best and underline the second-best result in each column.

Method	Identity Preserveness (↑)	Style Controllability (↑)	Creativeness (↑)	Overall Musicality (↑)
SongEcho [11]	2.84 ± 0.19	2.84 ± 0.23	2.56 ± 0.21	2.28 ± 0.21
ACE-Step-1.5 [12]	$\underline{3.53 \pm 0.21}$	2.93 ± 0.23	3.03 ± 0.21	3.31 ± 0.22
Suno-v5.5 [53]	$\mathbf{3.81 \pm 0.19}$	$\mathbf{3.72 \pm 0.21}$	$\underline{3.51 \pm 0.20}$	$\mathbf{3.88 \pm 0.21}$
MuLaCover	3.45 ± 0.20	$\underline{3.71 \pm 0.25}$	$\mathbf{3.65 \pm 0.22}$	$\underline{3.75 \pm 0.22}$

and thus reflects drift from the source rather than controlled stylization. This further demonstrates the flexibility of our control scheme.

4.4 Subjective Evaluation

Listening study. We complement the objective evaluation with a blind listening study. From the test set introduced in Section 4.3 we randomly select 8 source songs, each paired with a target style tag and the corresponding lyrics. For each song we generate a cover with all four systems (MuLaCover, ACE-Step-1.5, SongEcho, and Suno-v5.5), shuffle their order, and ask each rater to score every cover on four 5-point Likert scales: *Identity Preserveness*, *Style Controllability*, *Creativeness*, and *Overall Musicality*. Each participant was assigned a random subset of 5 of the 8 songs. We received 34 saved submissions. After excluding submissions with no complete case and dropping incomplete case blocks, the analysis retained 22 participants self-reporting their musical background as amateur (11), intermediate (6), or professional (5), yielding 95 complete participant–case blocks and 1,520 ratings in total. Partial completion resulted in 4.32 complete cases per retained participant on average.

Table 3 reports the per-metric mean and 95% confidence interval across all (participant, song) pairs. Under paired block-level Wilcoxon tests, MuLaCover significantly outperforming open-sourced baselines on Style Controllability, Creativeness, and Overall Musicality and is on par with Suno-v5.5. On Identity Preserveness, MuLaCover significantly outperforms SongEcho and shows no statistically significant difference from ACE-Step-1.5, while Suno-v5.5 remains significantly higher. The full questionnaire instrument, scoring rubrics, and detailed instructions are detailed in Appendix E.

Expert interview. The Likert questionnaire reveals *how good* each system is but not *why*; to probe the latter, we run a structured expert-interview protocol and report an R&B case study. We first sample one genre uniformly at random from a catalogue of common popular-music styles and, without playing any audio, ask the expert to enumerate from prior experience the salient attributes a faithful instance of that genre should exhibit, each with a short justification. We then sample one source song from the test set, present the four blinded covers, and ask the expert to rate and comment on each version against the elicited attributes. Finally, the commentary is fed to a language model to produce a concise summary verdict. The full protocol, the sampled genre, the elicited attributes, the organized expert record, and the summarization prompt are reproduced in Appendix F, where Versions A, B, C, D in the blinded record correspond to ACE-Step-1.5, SongEcho, MuLaCover, and Suno-v5.5, respectively. The summary verdict reads as follows:

> *Under the expert's two pre-specified R&B criteria—rhythm/groove and vocal melodic development—the four blinded versions rank D > C > B > A, with A deviating most toward electronic pop, B and C progressively closer to R&B's cyclic vocal phrasing, and D judged most canonical thanks to its natural groove and pop-R&B vocal timbre, melismas, and phrase endings.*

4.5 Ablation Studies

To isolate the contribution of each design choice in MuLaCover, we ablate three axes of the system in turn: (i) the *training schedule*, (ii) the *condition signal*, and (iii) the *architecture* used to inject that signal into the backbone. The metrics used in ablation studies share the same definition as described in Section 4.3 and are reported in Table 4. Due to limited compute, we retrain on a 70k-song subset (denoted MuLaCover-small) and use the same data scale for every variant.

Table 4: Ablation studies on the full-song cover task. The first row is the full MuLaCover; rows below ablate one component at a time, grouped by *training schedule*, *condition signal*, and *architecture*. ↑ / ↓ indicate higher / lower is better.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Variant	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
MuLaCover-small	8.28	6.36	7.67	7.89	4.58	4.56	4.33	4.42	4.45	0.14	0.38	0.49	0.98	1.04
Training schedule														
w/o Warmup	8.07	6.13	7.44	7.72	4.41	4.40	4.13	4.24	4.23	0.16	0.40	0.54	0.11	9.75
w/ SP	8.24	6.43	7.63	7.87	4.57	4.55	4.33	4.41	4.44	0.16	0.35	0.45	0.98	1.04
Condition signal														
w/o BPM-norm	8.21	6.36	7.61	7.84	4.56	4.55	4.33	4.40	4.43	0.14	0.36	0.51	0.73	2.70
w/o BPM-norm (SP.)	8.19	6.40	7.59	7.82	4.48	4.45	4.23	4.30	4.35	0.18	0.34	0.47	0.99	1.04
w/ F_0	8.10	6.46	7.55	7.74	4.45	4.41	4.19	4.24	4.41	0.20	0.39	0.52	0.33	5.20
w/ F_0 (SP.)	8.12	6.45	7.56	7.76	4.44	4.38	4.16	4.23	4.39	0.20	0.38	0.51	0.40	5.55
w/o Chord	7.97	6.30	7.39	7.65	4.35	4.32	4.07	4.16	4.16	0.20	0.38	0.50	0.46	4.86
Architecture														
w/o Gate	8.26	6.38	7.62	7.88	4.59	4.58	4.34	4.44	4.46	0.14	0.38	0.52	0.86	1.74
w/ Prefix tuning	7.97	5.81	7.37	7.67	4.34	4.35	4.08	4.16	4.18	0.43	0.36	0.58	0.09	11.58

Training schedule. We first ablate the short-clip warmup (Stage 1). The variant w/o Warmup skips Stage 1 and starts training directly from Stage 2, with one extra epoch in Stage 2 to match the total number of training steps. Without the warmup, the newly introduced modules fail to align with the backbone and c is not effectively injected. The output is no longer identifiable as a cover, and the broken injection collaterally drags down perceptual quality. We next ablate the cross-cover pairing in Stage 3, where c is drawn from a different rendition than the one supplying ℓ, s, a. The variant w/ SP (self-pair) reverts Stage 3 to the same-rendition setup of Stages 1 and 2, with extra epochs compensating for the reduced number of (c, ℓ, s, a) tuples. Style controllability deteriorates while everything else is preserved, confirming that the cross-cover stage is what unlocks the model's stylistic flexibility.

Condition signal. Our symbolic tonal-content signal is BPM-normalized to a canonical $\mathrm{BPM}{=}120$, with the intent of relaxing the otherwise strict frame-wise correspondence between c and the audio so as to enhance style controllability without compromising content control. To verify this, we replace c with a non-BPM-normalized symbolic sequence whose length and frame rate match the audio token stream, and train two variants: w/o BPM-norm (the full three-stage schedule including the cross-cover Stage 3) and w/o BPM-norm (SP.) (Stage 3 restricted to self-pairs). The two together expose a content/style tradeoff that is structural rather than incidental. Once c is strictly frame-wise aligned with the audio, the self-pair regime is fully consistent with the alignment and yields strong cover identifiability at the cost of style flexibility, whereas the cross-cover regime pairs the condition of one rendition with the audio of another, and a frame-wise signal becomes too brittle to small tempo or duration mismatches, so cover identifiability collapses instead. BPM normalization loosens the temporal alignment just enough to keep the content needed for cover identifiability while leaving the slack required by style control.

We note that SongEcho [11] uses a frame-wise F_0 sequence as its content control. Substituting our symbolic signal with the same F_0 representation gives the analogous pair of variants w/ F_0 and w/ F_0 (SP.), and the same content/style tradeoff between the two reappears. Beyond the tradeoff, F_0 based injection is uniformly weaker than the symbolic counterpart, which we attribute to two intrinsic properties of F_0: being continuous rather than discretized to per-pitch values, it is harder to internalize after embedding and thus less amenable to injection; and carrying only the vocal melody, it is a sparser and less informative signal in the first place.

The variant w/o Chord drops the chord progression and keeps only the melody, and ineffective injection emerges. In popular music the chord progression carries the harmonic *skeleton* of the song and is far more informative than the vocal melody alone, and we believe removing it leaves c too sparse for the model to reliably exploit.

Taken together, these results justify our choice of a BPM-normalized symbolic lead sheet (melody plus chord progression) as the tonal-content signal: among the design alternatives we considered, it best balances content control (cover identifiability) and style control.

Architecture. We first remove the learned gate g_l in Equation 3, reducing the injection to a plain residual cross-attention $\tilde{h}_l = h_l + \mathrm{CA}_l(h_l, c_l)$; we denote this variant w/o Gate. Cover identifiability suffers, consistent with our hypothesis that the adaptive gate is precisely the mechanism through

which the model selects which positions and feature dimensions of the conditioning update should be admitted into h_l. Without it, the symbolic information is broadcast indiscriminately and the model loses the freedom to bind the condition where it is actually useful. We next replace cross-attention-based injection with a prefix-tuning mechanism, previously shown to be effective for adapting LM-based music generators to new controls [26, 28]; we denote this variant w/ Prefix tuning. Here condition injection essentially fails, and prefix tuning even disrupts the lyrics-to-music mapping already established by the foundation model, as reflected in the PER blow-up. We attribute this to the fact that our backbone already consumes lyrics and style as prefix prompts, and forcing an additional newly introduced control interferes with the autoregressive mapping the model previously learned.

Overall, GACA is a sound and novel choice for condition injection on top of an LM-based music foundation model, compared with the alternatives we examined.

5 Discussion and Limitations

MuLaCover formulates cover song generation as lead-sheet-controlled music generation. However, we acknowledge that the notion of a cover song is inherently open-ended: what should remain invariant across versions can be defined at different granularities, ranging from strictly preserving the lead sheet, to allowing re-harmonization in a jazz arrangement, to borrowing only a few motifs from the original as inspiration. Our current formulation fixes this boundary at the lead-sheet level. To better support human-AI co-creation in real-world scenarios, future work could offer more intuitive controls that let users adjust how closely the generated song adheres to the original.

Copyright protection for AI-generated content remains an open concern, and it is especially relevant to cover song generation, which by nature takes existing songs as input. We respect the intellectual property of human artists. All training data used in this work are synthetic and were obtained from music generation providers in accordance with their terms of service. Since generated covers may constitute derivative works of the original songs, users are responsible for obtaining the necessary permissions from rights holders before distributing or commercially using such outputs.

6 Conclusion

We present MuLaCover, a lead-sheet-conditioned song generation system for flexible cover generation that conditions a pretrained text-to-song foundation model on a BPM-normalized symbolic lead sheet via a gated cross-attention module. MuLaCover is alignment-free, supporting flexible tempo and partial-reference generation in a single unified model. Objective metrics, listening-study questionnaires, and a structured expert-interview protocol jointly show that MuLaCover outperforms open-source cover baselines and is broadly on par with the leading closed-source commercial system Suno-v5.5.

Acknowledgments and Disclosure of Funding

We thank Nanjing University of the Arts for helping recruit participants for the subjective evaluation. This work was done while Lynn Yi was an intern at Vera Praxis Lab, which also provided the GPU resources used in this work. This work is supported by the National Natural Science Foundation of China (62376011) and the National Key R&D Program of China (2024YFA1410000).

References

[1] Joan Serrà, Emilia Gómez, and Perfecto Herrera. *Audio Cover Song Identification and Similarity: Background, Approaches, Evaluation, and Beyond*, pages 307–332. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-11674-2. doi: 10.1007/978-3-642-11674-2_14. URL https://doi.org/10.1007/978-3-642-11674-2_14.

[2] Daniel P.W. Ellis and Graham E. Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 4, pages IV–1429–IV–1432, 2007. doi: 10.1109/ICASSP.2007.367348.

[3] Joan Serra, Emilia Gomez, Perfecto Herrera, and Xavier Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(6):1138–1151, 2008. doi: 10.1109/TASL.2008.924595.

[4] Zhesong Yu, Xiaoshuo Xu, Xiaoou Chen, and Deshun Yang. Learning a representation for cover song identification using convolutional neural network. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 541–545, 2020. doi: 10.1109/ICASSP40776.2020.9053839.

[5] Xingjian Du, Zhesong Yu, Bilei Zhu, Xiaoou Chen, and Zejun Ma. Bytecover: Cover song identification via multi-loss training. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 551–555, 2021. doi: 10.1109/ICASSP39728.2021.9414128.

[6] Xingjian Du, Zijie Wang, Xia Liang, Huidong Liang, Bilei Zhu, and Zejun Ma. Bytecover3: Accurate cover song identification on short queries. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10095389.

[7] Feng Liu, Deyi Tuo, Yinan Xu, and Xintong Han. CoverHunter: Cover Song Identification with Refined Attention and Alignments . In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1080–1085, Los Alamitos, CA, USA, July 2023. IEEE Computer Society. doi: 10.1109/ICME55011.2023.00189. URL https://doi.ieeecomputersociety.org/10.1109/ICME55011.2023.00189.

[8] Ruibin Yuan, Hanfeng Lin, Shuyue Guo, Ge Zhang, Jiahao Pan, Yongyi Zang, Haohe Liu, Yiming Liang, Wenye Ma, Xingjian Du, et al. Yue: Scaling open foundation models for long-form music generation. *arXiv preprint arXiv:2503.08638*, 2025.

[9] Junmin Gong, Sean Zhao, Sen Wang, Shengyuan Xu, and Joe Guo. Ace-step: A step towards music generation foundation model. *arXiv preprint arXiv:2506.00045*, 2025.

[10] Dongchao Yang, Yuxin Xie, Yuguo Yin, Zheyu Wang, Xiaoyu Yi, Gongxi Zhu, Xiaolong Weng, Zihan Xiong, Yingzhe Ma, Dading Cong, et al. Heartmula: A family of open sourced music foundation models. *arXiv preprint arXiv:2601.10547*, 2026.

[11] Sifei Li, Yang Li, Zizhou Wang, Yuxin Zhang, Fuzhang Wu, Oliver Deussen, Tong-Yee Lee, and Weiming Dong. Songecho: Towards cover song generation via instance-adaptive element-wise linear modulation. In *The Fourteenth International Conference on Learning Representations*.

[12] Junmin Gong, Yulin Song, Wenxiao Zhao, Sen Wang, Shengyuan Xu, Jing Guo, and Xuerui Yang. Ace-step 1.5: Pushing the boundaries of open-source music generation. *arXiv preprint arXiv:2602.00744*, 2026.

[13] Ye Bai, Haonan Chen, Jitong Chen, Zhuo Chen, Yi Deng, Xiaohong Dong, Lamtharn Hantrakul, Weituo Hao, Qingqing Huang, Zhongyi Huang, et al. Seed-music: A unified framework for high quality and controlled music generation. *arXiv preprint arXiv:2409.09214*, 2024.

[14] Alexandre Papadopoulos, Pierre Roy, and François Pachet. Assisted lead sheet composition using flowcomposer. In Michel Rueher, editor, *Principles and Practice of Constraint Programming*, pages 769–785, Cham, 2016. Springer International Publishing. ISBN 978-3-319-44953-1.

[15] Hao-Min Liu and Yi-Hsuan Yang. Lead sheet generation and arrangement by conditional generative adversarial network. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 722–727, 2018. doi: 10.1109/ICMLA.2018.00114.

[16] Ziqian Ning, Huakang Chen, Yuepeng Jiang, Chunbo Hao, Guobin Ma, Shuai Wang, Jixun Yao, and Lei Xie. Diffrhythm: Blazingly fast and embarrassingly simple end-to-end full-length song generation with latent diffusion. *arXiv preprint arXiv:2503.01183*, 2025.

[17] Shun Lei, Yaoxun Xu, Huaicheng Zhang, Hangting Chen, Yixuan Zhang, Chenyu Yang, Haina Zhu, Shuai Wang, Zhiyong Wu, Dong Yu, et al. Levo: High-quality song generation with multi-preference alignment. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[18] Wei-Ho Tsai, Hung-Ming Yu, Hsin-Min Wang, et al. Query-by-example technique for retrieving cover versions of popular songs with similar melodies. In *ISMIR*, volume 5, pages 183–190, 2005.

[19] Özgür İzmirli. Tonal similarity from audio using a template based attractor model. In *ISMIR*, pages 540–545, 2005.

[20] Mark A Bartsch and Gregory H Wakefield. To catch a chorus: Using chroma-based representations for audio thumbnailing. In *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No. 01TH8575)*, pages 15–18. IEEE, 2001.

[21] Emilia Gómez. Tonal description of music audio signals. *Department of Information and Communication Technologies*, 2006.

[22] Furkan Yesiler, Joan Serrà, and Emilia Gómez. Accurate and scalable version identification using musically-motivated embeddings. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–25, 2020. doi: 10.1109/ICASSP40776.2020.9053793.

[23] Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, et al. Musiclm: Generating music from text. *arXiv preprint arXiv:2301.11325*, 2023.

[24] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. *Advances in neural information processing systems*, 36:47704–47720, 2023.

[25] Shun Lei, Yixuan Zhou, Boshi Tang, Max W Lam, Feng Liu, Hangyu Liu, Jingcheng Wu, Shiyin Kang, Zhiyong Wu, and Helen Meng. Songcreator: Lyrics-based universal song generation. *Advances in Neural Information Processing Systems*, 37:80107–80140, 2024.

[26] Liwei Lin, Gus Xia, Junyan Jiang, and Yixiao Zhang. Content-based controls for music large language modeling. *arXiv preprint arXiv:2310.17162*, 2023.

[27] Fang-Duo Tsai, Shih-Lun Wu, Weijaw Lee, Sheng-Ping Yang, Bo-Rui Chen, Hao-Chung Cheng, and Yi-Hsuan Yang. Musecontrollite: Multifunctional music generation with lightweight conditioners. In *Forty-second International Conference on Machine Learning*.

[28] Yun-Han Lan, Wen-Yi Hsiao, Hao-Chung Cheng, and Yi-Hsuan Yang. Musicongen: Rhythm and chord control for transformer-based text-to-music generation. *arXiv preprint arXiv:2407.15060*, 2024.

[29] Peiling Lu, Xin Xu, Chenfei Kang, Botao Yu, Chengyi Xing, Xu Tan, and Jiang Bian. Musecoco: Generating symbolic music from text. *CoRR*, abs/2306.00110, 2023. doi: 10.48550/ARXIV.2306.00110. URL `https://doi.org/10.48550/arXiv.2306.00110`.

[30] Hao-Wen Dong, Ke Chen, Shlomo Dubnov, Julian J. McAuley, and Taylor Berg-Kirkpatrick. Multitrack music transformer. In *ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023*, pages 1–5. IEEE, 2023. doi: 10.1109/ICASSP49357.2023.10094628. URL `https://doi.org/10.1109/ICASSP49357.2023.10094628`.

[31] Dimitri von Rütte, Luca Biggio, Yannic Kilcher, and Thomas Hofmann. FIGARO: controllable music generation using learned and expert features. In *ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=NyR8OZFHw6i.

[32] Ziyu Wang, Dingsu Wang, Yixiao Zhang, and Gus Xia. Learning interpretable representation for controllable polyphonic music generation. In *ISMIR 2020, Montreal, Canada, October 11-16, 2020*, pages 662–669, 2020. URL http://archives.ismir.net/ismir2020/paper/000094.pdf.

[33] Lejun Min, Junyan Jiang, Gus Xia, and Jingwei Zhao. Polyffusion: A diffusion model for polyphonic score generation with internal and external controls. In *ISMIR 2023, Milan, Italy, November 5-9, 2023*, pages 231–238, 2023. doi: 10.5281/ZENODO.10265265. URL https://doi.org/10.5281/zenodo.10265265.

[34] Xiaoyu Yi, Qi He, Gus Xia, and Ziyu Wang. Vitex: Visual texture control for multi-track symbolic music generation via discrete diffusion models. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 14797–14801. IEEE, 2026.

[35] Ziyu Wang, Lejun Min, and Gus Xia. Whole-song hierarchical generation of symbolic music using cascaded diffusion models. In *ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=sn7CYWyavh.

[36] 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.

[37] Jingwei Zhao, Ziyu Wang, Gus Xia, and Ye Wang. Bossa: Learning music style through cross-modal bootstrapping. In *AI for Music Workshop*.

[38] Chih-Pin Tan, Shuen-Huei Guan, and Yi-Hsuan Yang. Picogen: Generate piano covers with a two-stage approach. In *Proceedings of the 2024 International Conference on Multimedia Retrieval*, pages 1180–1184, 2024.

[39] Fang-Duo Tsai, Yi-An Lai, Fei-Yueh Chen, Hsueh-Wei Fu, Li Chai, Wei-Jaw Lee, Hao-Chung Cheng, and Yi-Hsuan Yang. MIDI-informed singing accompaniment generation in a compositional song pipeline. *arXiv preprint arXiv:2602.22029*, 2026.

[40] Qi He, Ziyu Wang, and Gus Xia. TOMI: transforming and organizing music ideas for multi-track compositions with full-song structure. In Juhan Nam, Dasaem Jeong, Keunwoo Choi, Li Su, Magdalena Fuentes, Tomoyasu Nakano, Xiao Hu, and Hao-Wen (Herman) Dong, editors, *Proceedings of the 26th International Society for Music Information Retrieval Conference, ISMIR 2025, Daejeon, South Korea, September 21-25, 2025*, pages 337–345, 2025. doi: 10.5281/ZENODO.17811376. URL https://doi.org/10.5281/zenodo.17811376.

[41] Sungkyun Chang, Emmanouil Benetos, Holger Kirchhoff, and Simon Dixon. Yourmt3+: Multi-instrument music transcription with enhanced transformer architectures and cross-dataset stem augmentation. In *2024 IEEE 34th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2024.

[42] Junyan Jiang, Ke Chen, Wei Li, and Gus Xia. Large-vocabulary chord transcription via chord structure decomposition. In *ISMIR*, pages 644–651, 2019.

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

[44] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[45] Changhao Jiang, Jiahao Chen, Zhenghao Xiang, Zhixiong Yang, Hanchen Wang, Jiabao Zhuang, Xinmeng Che, Jiajun Sun, Hui Li, Yifei Cao, et al. Muse: Towards reproducible long-form song generation with fine-grained style control. *arXiv preprint arXiv:2601.03973*, 2026.

[46] Alexandre Défossez, Nicolas Usunier, Léon Bottou, and Francis Bach. Demucs: Deep extractor for music sources with extra unlabeled data remixed. *arXiv preprint arXiv:1909.01174*, 2019.

[47] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[48] Taejun Kim and Juhan Nam. All-in-one metrical and functional structure analysis with neighborhood attentions on demixed audio. In *2023 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 1–5. IEEE, 2023.

[49] Chunbo Hao, Ruibin Yuan, Jixun Yao, Qixin Deng, Xinyi Bai, Wei Xue, and Lei Xie. Songformer: Scaling music structure analysis with heterogeneous supervision. *arXiv preprint arXiv:2510.02797*, 2025.

[50] Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, et al. Qwen2-audio technical report. *arXiv preprint arXiv:2407.10759*, 2024.

[51] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[52] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

[53] Mar 2026. URL `https://suno.com/blog/v5-5`.

[54] Andros Tjandra, Yi-Chiao Wu, Baishan Guo, John Hoffman, Brian Ellis, Apoorv Vyas, Bowen Shi, Sanyuan Chen, Matt Le, Nick Zacharov, et al. Meta audiobox aesthetics: Unified automatic quality assessment for speech, music, and sound. *arXiv preprint arXiv:2502.05139*, 2025.

[55] Jixun Yao, Guobin Ma, Huixin Xue, Huakang Chen, Chunbo Hao, Yuepeng Jiang, Haohe Liu, Ruibin Yuan, Jin Xu, Wei Xue, et al. Songeval: A benchmark dataset for song aesthetics evaluation. *arXiv preprint arXiv:2505.10793*, 2025.

[56] Haina Zhu, Yizhi Zhou, Hangting Chen, Jianwei Yu, Ziyang Ma, Rongzhi Gu, Yi Luo, Wei Tan, and Xie Chen. Muq: Self-supervised music representation learning with mel residual vector quantization. *IEEE Transactions on Audio, Speech and Language Processing*, 2025.

[57] Longshen Ou, Jingwei Zhao, Ziyu Wang, Gus Xia, Qihao Liang, and Torin Hopkins Ye Wang. Unifying symbolic music arrangement: Track-aware reconstruction and structured tokenization. *arXiv preprint arXiv:2408.15176*, 2024.

Appendix

A Stage 3 Cross-Cover Data Pipeline

We prompt an LLM to generate $1{,}250$ lyric sets paired with HeartMuLa-format style tags (also LLM-generated), and feed them to music generation providers to synthesize $1{,}250$ source songs. For each source song, we then feed its audio, its lyrics, and seven additional, pairwise-distinct LLM-generated style tags to music generation providers to produce seven cover versions; Because each music generation provider returns two songs, the procedure above is effectively doubled, yielding a cross-cover dataset of $20{,}000$ paired covers.

For training, we form pairs by sampling two distinct covers of the same source song. Our network is sensitive to global key, so a cover whose key has shifted during cover generation is unusable for training. We therefore extract lead sheets from both covers in each candidate pair, run the key detection of [57], and discard pairs whose keys disagree. From only $20{,}000$ songs this yields roughly $130{,}000$ trainable pairs, substantially improving data utilization and strengthening the style controllability of the model.

B Per-Stage Training Hyperparameters

Table 5 reports the per-stage training hyperparameters.

Table 5: Per-stage training hyperparameters. GACA: gated adaptive cross-attention. Entries marked $\star$ follow the dynamic mask schedule described below.

Hyperparameter	Stage 1	Stage 2 (w/o mask)	Stage 2 (w/ mask)	Stage 3
Learning rate	2×10^{-4}	2×10^{-4}	2×10^{-4}	2×10^{-5}
Epochs	3	2	2	2
Avg. bs/GPU	9	1	1	1
(α, β, ρ)	$(0.0, 1.0, 0.0)$	$(0.0, 1.0, 0.0)$	$\star$	$\star$
Trainable modules	GACA	GACA	GACA	GACA & LoRA
Trainable params	$\sim$300M	$\sim$300M	$\sim$300M	$\sim$400M
Time per epoch	0.5 h	$\sim$5 h	$\sim$5 h	$\sim$2 h
Optimizer	AdamW			
LR scheduler	warmup followed by cosine annealing			
Gradient accum.	4			
GPUs	16			

Mask configuration (α, β, ρ)**.** α, β, and ρ denote the start ratio, end ratio, and invert ratio of the masked window. For Stage 1 and Stage 2 (w/o mask), the configuration $(0.0, 1.0, 0.0)$ corresponds to the default full-song setting. For entries marked $\star$ (Stage 2 w/ mask and Stage 3), we fix $\rho = 0.5$ and sample α, β per call: a window-length ratio r is either fixed or drawn from a curriculum range $r \sim \mathcal{U}(r_{\min}, r_{\max})$; then $\alpha \sim \mathcal{U}(0, 1-r)$ and $\beta = \alpha + r$.

C Task Setup and Baseline Configurations

The partial-reference cover task (Task 2; Section 4.2) exposes only a fragment of the reference recording to the model. For each of the 32 reference pieces $i = 1, 2, \ldots, 32$, we manually label the start and end α_i, β_i of the first chorus. The three setups—continuation, outpainting, and inpainting—are then specified by instantiating the active-position mask $\mathcal{A}(\alpha, \beta, \rho)$ from Eq. 5 as follows. *(i)* **Continuation** $[\mathcal{A}(0, \alpha_i, 0)]$: the prefix from the song's start through the section preceding the first chorus is observed and the remainder is generated. *(ii)* **Outpainting** $[\mathcal{A}(\alpha_i, \beta_i, 0)]$: only the first chorus is observed and the rest of the song (preceding and following) is generated outward from it. *(iii)* **Inpainting** $[\mathcal{A}(\alpha_i, \beta_i, 1)]$: everything except the first chorus is observed, and the model is asked to fill in the missing chorus.

For each configuration, the observed fragment supplies the tonal-content condition for the corresponding region, while the unobserved region is generated freely under the target lyrics and style tag.

For SongEcho [11], we refer to its official implementation[3]. For inference on Task 1, we extract the f0 sequence using its provided RMVPE model and follow its official inference Python script. We use its released checkpoint with all other parameters set to default. For Task 2, the SongEcho paper claims support for inpainting and outpainting, also showcased on their demo page[4]; however, the relevant instructions and inference script for this setting are not found in their open-sourced code. We therefore refer to its description in the original paper, which we quote here: "Our method supports inpainting and outpainting via a simple masking strategy" (Sec. 5.3.2 of the SongEcho paper), and implement a masking strategy based on our understanding: for a given reference music, the observed part is kept as the condition, and the rest is masked out as zero silence.

For ACE-Step 1.5 [12], we refer to its official implementation[5]. We use its turbo version. We refer to its official documentation for cover song generation[6]. Besides lyrics and style prompts, the influence of the reference music is controlled by the parameter `audio_cover_strength`. When this parameter is set too high (≥ 0.4), we empirically find that the model copies the original song; when set too low (≤ 0.2), it cannot preserve the tonal content and therefore cannot be identified as a cover given the reference. To ensure a fair comparison and a genuine baseline performance, we set `audio_cover_strength` to 0.3. We additionally report ACE-Step 1.5's objective metrics in Table 6 while iterating through `audio_cover_strength` λ from 0.1 to 0.8. All the metrics converge as $\lambda \geq 0.4$, with 100 percent accuracy for cover song identification model. It is exactly caused by the model copying the original song and producing identical inference results. While as $\lambda = 0.1$ or 0.2, the accuracy of cover song identification is low, indicating that the model can not preserve the necessary tonal content that a cover version should keep invariant. For Task 2, ACE-Step 1.5 does not natively support our setting; however, this can be achieved by cascading its cover and repaint features. Cascading approach: for a reference music, split it into two parts—the partial-condition part and the remaining part. For the partial-condition part, we separately use the cover feature to generate a cover, and we substitute the original portion with the generated part. We then feed the whole song (partial-condition part replaced by the cover result while the remaining stays the same) into the model and ask it to repaint the rest.

Table 6: Objective evaluation results under different audio cover control strengths λ for ACE-Step 1.5.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
ACE-Step-1.5 $\lambda = 0.1$	8.25	5.58	7.64	7.80	4.21	4.17	3.93	4.08	4.07	0.37	0.39	0.56	0.08	12.40
ACE-Step-1.5 $\lambda = 0.2$	8.23	5.87	7.64	7.78	4.28	4.25	4.03	4.14	4.17	0.29	0.37	0.50	0.25	8.03
ACE-Step-1.5 $\lambda = 0.3$	8.28	6.30	7.67	7.86	4.48	4.46	4.25	4.34	4.36	0.21	0.30	0.40	0.95	1.10
ACE-Step-1.5 $\lambda = 0.4$	8.28	6.40	7.64	7.87	4.51	4.49	4.29	4.37	4.38	0.13	0.27	0.36	1.00	1.00
ACE-Step-1.5 $\lambda = 0.5$	8.27	6.43	7.61	7.86	4.51	4.50	4.29	4.38	4.38	0.11	0.27	0.34	1.00	1.00
ACE-Step-1.5 $\lambda = 0.6$	8.27	6.42	7.61	7.85	4.51	4.49	4.28	4.38	4.38	0.12	0.27	0.34	1.00	1.00
ACE-Step-1.5 $\lambda = 0.7$	8.25	6.43	7.60	7.84	4.51	4.49	4.28	4.38	4.38	0.13	0.27	0.33	1.00	1.00
ACE-Step-1.5 $\lambda = 0.8$	8.25	6.44	7.59	7.83	4.51	4.50	4.27	4.38	4.37	0.12	0.27	0.33	1.00	1.00

Table 7: Partial-reference cover, *continuation* configuration. Same metrics as Table 1.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
SongEcho [11]	7.49	5.44	6.63	7.43	3.06	2.98	2.76	2.77	2.83	0.64	0.19	0.70	0.25	7.59
ACE-Step-1.5 [12]	8.26	5.86	7.65	7.83	4.19	4.17	3.91	4.05	4.04	0.36	0.35	0.51	0.38	6.18
MuLaCover	8.33	6.33	7.71	7.93	4.59	4.59	4.33	4.44	4.47	0.11	0.42	0.51	0.63	3.15

D Partial-Reference Cover: Per-Configuration Results

Table 2 in the main text reports objective scores averaged over the three masking configurations. Tables 7–9 report the per-configuration breakdown. Among each configuration, MuLaCover achieves superior results over baselines.

[3] https://github.com/lsfhuihuiff/SongEcho_ICLR2026

[4] https://vvanonymousvv.github.io/SongEcho_updated/

[5] https://github.com/ace-step/ACE-Step-1.5

[6] https://github.com/ace-step/ACE-Step-1.5/blob/main/docs/en/INFERENCE.md#2-cover

Table 8: Partial-reference cover, *outpainting* configuration. Same metrics as Table 1.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
SongEcho [11]	7.41	5.43	6.56	7.32	2.87	2.76	2.63	2.58	2.67	0.64	0.20	0.72	0.25	7.61
ACE-Step-1.5 [12]	8.23	5.71	7.62	7.82	4.17	4.13	3.89	4.02	4.01	0.40	0.36	0.53	0.38	6.89
MuLaCover	8.31	6.27	7.66	7.91	4.54	4.54	4.29	4.39	4.43	0.12	0.42	0.51	0.55	3.36

Table 9: Partial-reference cover, *inpainting* configuration. Same metrics as Table 1.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
SongEcho [11]	7.83	5.90	7.20	7.56	3.50	3.37	3.17	3.27	3.24	0.38	0.35	0.46	0.73	2.62
ACE-Step-1.5 [12]	8.30	6.16	7.70	7.87	4.31	4.30	4.06	4.16	4.19	0.32	0.32	0.46	0.82	1.68
MuLaCover	8.34	6.33	7.70	7.93	4.58	4.58	4.33	4.43	4.46	0.13	0.41	0.49	0.93	1.28

E Subjective Evaluation: Listening Study

Protocol. On each rating page the participant is presented with the reference music, the target style tag, and the target lyrics, and is asked to score the cover on four 5-point Likert scales: *Identity Preserveness*, *Style Controllability*, *Creativeness*, and *Overall Musicality*. The verbatim wording of each metric and the anchor description for each of the scores 1–5 are shown on the case page in Figure 2. Before the rating phase the participant reports age and self-assessed musical background (amateur, intermediate, or professional). Participants could stop at any point, with completed responses auto-saved, so as to mitigate listening fatigue. All participants in both the listening study and the expert interview were unpaid volunteers and gave informed consent on the welcome page before starting the questionnaire.

Participants. We received 34 saved submissions. After excluding 12 submissions with no complete case and dropping incomplete case blocks, the analysis retained 22 participants self-reporting their musical background as amateur (11), intermediate (6), or professional (5). Among participants with valid age responses (n=18), age was 25.5 ± 4.1 (range 21–33). Each retained participant completed 4.32 cases on average, yielding 95 complete (participant, case) blocks and 1,520 ratings in total. A complete block is a participant–case tuple in which all four systems were rated on all four metrics. Table 10 breaks the average rating down by self-reported musical background, pooled across systems.

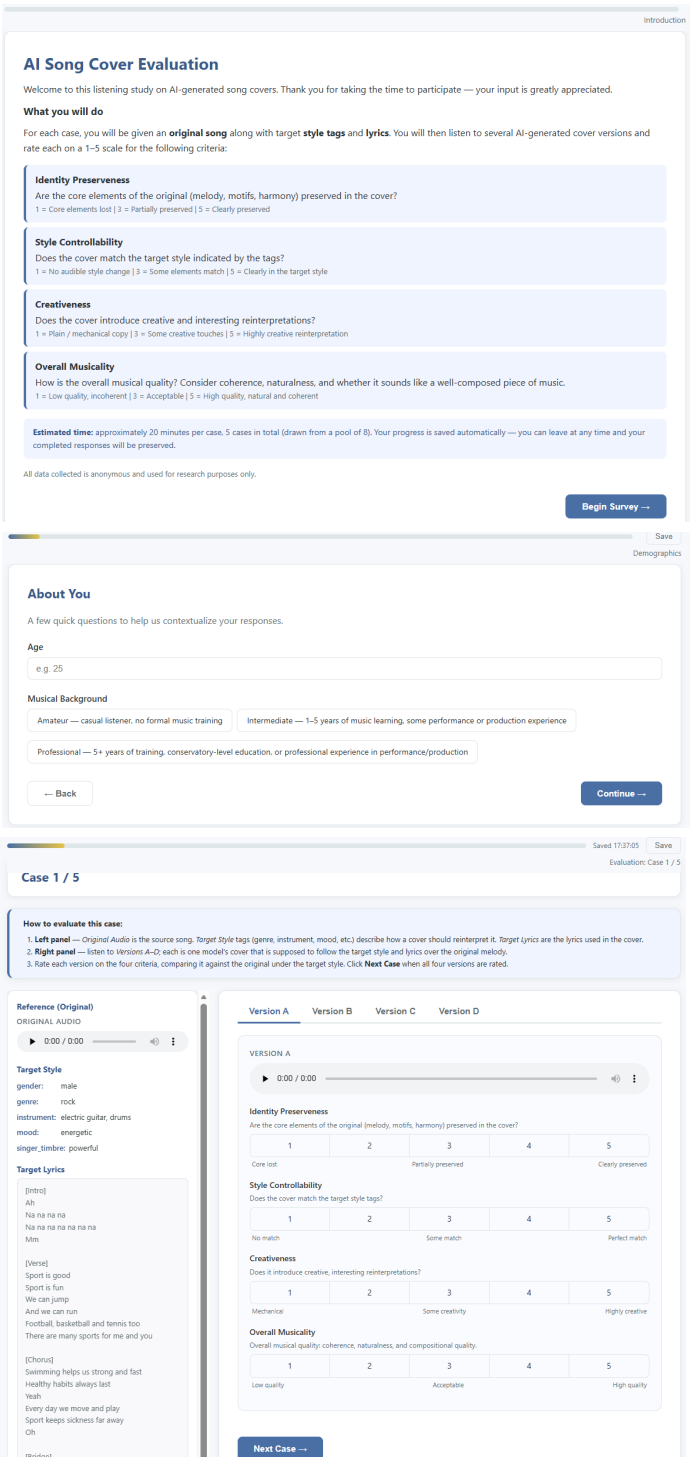


Figure 2: Listening-study questionnaire interface. Top: welcome page. Middle: demographics page. Bottom: per-case rating page, listing the four metrics together with the textual anchors for scores 1–5.

Omnibus test across systems. Because every (participant, case) block receives one rating from each of the four systems on each metric, the rating data is within-subject and paired. We therefore use the complete participant–case tuple as the paired analysis block: for each metric, the $k{=}4$ system ratings are ranked within each of the $n_{\text{blocks}}{=}95$ blocks and tested for system-level rank differences

Table 10: Mean rating per metric by self-reported musical background, pooled across systems, with cluster-bootstrap 95% CI in brackets (resampling at the participant level, 1,000 replicates). n is the number of (participant, case, system) ratings contributing to each row, per metric.

Background	n	Identity Preserveness	Style Controllability	Creativeness	Overall Musicality
Amateur	200	3.54 [3.27, 3.79]	3.38 [3.12, 3.62]	3.21 [3.00, 3.40]	3.50 [3.35, 3.67]
Intermediate	100	3.25 [2.54, 3.80]	3.08 [2.37, 3.70]	3.16 [2.69, 3.58]	3.26 [2.82, 3.64]
Professional	80	3.26 [2.82, 3.88]	3.36 [2.67, 4.05]	3.17 [2.67, 3.51]	2.89 [2.28, 3.34]

with a Friedman rank test. The 95 complete blocks come from 112 evaluated cases after dropping 17 incomplete cases, and each complete block contributes $4\times4=16$ ratings. Table 11 reports the test statistic, p-value, and Kendall's coefficient of concordance W as effect size. Significant system-level differences are observed on every one of the four metrics, with W ranging from 0.19 (Creativeness) to 0.31 (Overall Musicality), motivating the pairwise comparisons below. We report these p-values as paired block-level evidence of within-block system differences, rather than as evidence from 95 fully independent subject-level observations.

Table 11: Friedman omnibus test across the four systems, per metric. n_{blocks} is the number of (participant, case) blocks within which the $k=4$ system ratings are ranked. Kendall's W is the coefficient-of-concordance effect size.

Metric	n_{blocks}	χ^2	p-value	Kendall's W
Identity Preserveness	95	57.19	2.3×10^{-12}	0.20
Style Controllability	95	55.69	4.9×10^{-12}	0.20
Creativeness	95	52.90	1.9×10^{-11}	0.19
Overall Musicality	95	88.08	5.7×10^{-19}	0.31

Pairwise post-hoc tests. Conditional on the omnibus rejecting on every metric, we run pairwise Wilcoxon signed-rank tests on the within-block (participant, case) score differences for each of the $\binom{4}{2}=6$ system pairs, and apply a Bonferroni correction over the 6 comparisons within each metric. Table 12 reports the corrected p-values; bold entries are significant at the 0.05 level. Three observations follow. *(i)* On Style Controllability, Creativeness, and Overall Musicality, MuLaCover shows no significant difference from Suno-v5.5 ($p_{\text{bonf}}=1.0$ on all three) while significantly outperforming both ACE-Step-1.5 ($p_{\text{bonf}}\leq0.017$) and SongEcho ($p_{\text{bonf}}<10^{-4}$ on all three). *(ii)* On Identity Preserveness, MuLaCover significantly outperforms SongEcho ($p_{\text{bonf}}=0.0001$) and shows no significant difference from ACE-Step-1.5 ($p_{\text{bonf}}=1.0$), while Suno-v5.5 remains significantly higher than MuLaCover ($p_{\text{bonf}}=0.006$). *(iii)* SongEcho is significantly lower than MuLaCover and Suno-v5.5 on all four metrics; compared with ACE-Step-1.5, SongEcho is lower on Identity Preserveness, Creativeness, and Overall Musicality, but the Style Controllability difference is not significant.

Table 12: Bonferroni-corrected p-values from pairwise Wilcoxon signed-rank tests across the four systems, per metric. Each test pairs the two systems on the within-block (participant, case) score differences over the $n_{\text{blocks}}=95$ complete blocks; the Bonferroni correction is applied over the 6 pairs within each metric. Bold marks $p < 0.05$.

Pair	Identity	Style	Creativeness	Overall
ACE-Step-1.5 vs. MuLaCover	1.0000	$\mathbf{< 10^{-4}}$	**0.0004**	**0.0161**
ACE-Step-1.5 vs. SongEcho	$\mathbf{< 10^{-4}}$	1.0000	**0.0169**	$\mathbf{< 10^{-4}}$
ACE-Step-1.5 vs. Suno-v5.5	**0.0475**	$\mathbf{< 10^{-4}}$	**0.0007**	$\mathbf{< 10^{-4}}$
MuLaCover vs. SongEcho	**0.0001**	$\mathbf{< 10^{-4}}$	$\mathbf{< 10^{-4}}$	$\mathbf{< 10^{-4}}$
MuLaCover vs. Suno-v5.5	**0.0060**	1.0000	1.0000	1.0000
SongEcho vs. Suno-v5.5	$\mathbf{< 10^{-4}}$	$\mathbf{< 10^{-4}}$	$\mathbf{< 10^{-4}}$	$\mathbf{< 10^{-4}}$

F Subjective Evaluation: Expert Interviews

Protocol. The expert interview is designed to explain the Likert results without letting the evaluator choose criteria after hearing the generated audio. In each session, we first sample a target genre from a catalogue of common popular-music styles. Before any audio is played, the expert is asked to list the most salient musical attributes of that genre and briefly explain how each attribute should be judged. Only after these attributes are fixed do we present the blinded covers from the four systems. The expert then scores and comments on each version using the pre-elicited attributes as the evaluation criteria. Finally, we feed the organized expert record to a language model and ask it to produce a concise, neutral summary. The respondent identifier and date are anonymized in the record below.

R&B case study. The sampled genre in the recorded session is R&B. The four blinded versions were later decoded as Version A = ACE-Step-1.5, Version B = SongEcho, Version C = MuLaCover, and Version D = Suno-v5.5. The tables and notes below are translated from the original Chinese notes and lightly edited for readability while preserving the expert's recorded ratings and judgments.

Table 13: Primary R&B attributes elicited from the expert before any generated audio was played.

Attribute	Expert definition / judging criterion
Rhythm and groove	The most salient property of rhythm-and-blues: drums and bass jointly form the groove, and faithful R&B should exhibit typical rhythmic patterns rather than a generic pop or electronic feel.
Vocal melody, repetition, and variation	In R&B, melodic development often happens through modular repetition and varied repetition over a cyclic rhythmic/harmonic frame; phrase-end ornaments, melismas, and vocal timbre are important cues.

Table 14: Organized expert record for the R&B case study. Each attribute follows the pre-listening criteria in Table 13. The table focuses on the two primary attributes for which the expert gave complete scores.

Version / system	Rhythm and groove	Vocal melody / timbre	Overall comment
A / ACE-Step-1.5	2: Electric drums, 808, and synthetic bass make the version sound closer to electronic pop.	2: The vocal line is closer to a normal pop-song melody than to R&B melodic development.	The first two attributes deviate substantially from typical R&B; the version leans toward electronic pop rather than canonical R&B.
B / SongEcho	3: The drum timbre and groove contain pop-R&B characteristics.	3: The vocal development is closer to R&B: it uses a four-phrase loop rather than the linear emotional buildup typical of pop ballads.	Overall, this version is relatively better than Version A.
C / MuLaCover	4: The expert rated the rhythm and groove higher than Versions A and B.	4: The expert rated the vocal melodic treatment higher than Versions A and B; the verse also has a vocal timbre that sounds very R&B-like.	The expert judged it close to Version B in broad style but with a more R&B-like singer timbre; it is still not the most canonical R&B rendition.
D / Suno-v5.5	4.5: The groove is slightly better than the preceding versions.	4.5: The vocal line is much more natural and close to pop R&B, especially in phrase endings, melismas, and timbre.	The expert also noted that the auxiliary harmonic and instrumentation-related attributes were relatively well matched.

Table 15: Objective evaluation on the full song cover task (Task 1) on human-recordings. ↑ / ↓ indicate higher / lower is better. **Bold** marks the best.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style		CSI judge	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
SongEcho [11]	7.66	**6.26**	7.06	7.32	3.47	3.23	3.13	3.19	3.27	0.42	0.32	0.43	0.97	1.03
ACE-Step-1.5 [12]	8.27	6.08	7.67	7.86	4.37	4.36	4.15	4.25	4.25	0.25	0.31	0.31	0.98	1.08
MuLaCover	**8.34**	6.24	**7.73**	**7.92**	**4.52**	**4.51**	**4.27**	**4.37**	**4.40**	**0.12**	**0.40**	**0.45**	**0.98**	**1.02**

The expert also listed harmonic looping and instrumentation as auxiliary references, but described them as less diagnostic than rhythm/groove and vocal melodic treatment. We therefore report only the two primary criteria in Table 14. In the original notes, the auxiliary comments are sparse: ACE-Step-1.5 receives a harmony score of 3 without an additional written rationale, and Suno-v5.5 is described in the overall comment as relatively well matched on the harmonic and instrumentation-related cues.

Prompt used for the language-model summary. For step (iv), we used the following prompt template to convert the organized expert record into a concise verdict. To avoid first-person or system-ownership bias, the prompt asks the language model to reason over the blinded version labels only. The version labels were mapped back to system names after the summary was produced.

> **Input:** A structured expert-interview record for an AI song-cover evaluation. The record contains: (1) the sampled target genre; (2) the musical attributes elicited from the expert before any generated audio was played; and (3) blinded versions A–D with per-attribute scores and expert comments.
> **Task:** Produce a concise 4–6 sentence neutral research summary contrasting Versions A–D. Base the verdict only on the expert's stated attributes, scores, and comments. Mention the genre-specific criteria that drive the ranking. Do not add claims that are not supported by the expert record. Do not use first-person labels such as "our model" or infer that any version should be preferred from its label.
> **Desired output:** A short neutral paragraph contrasting Version A, Version B, Version C, and Version D.

G Generalization Ability on Human Recordings

MuLaCover is trained on synthetic data. To evaluate whether our approach generalizes beyond training distribution, we additionally conduct an experiment on real human recordings.

We manually choose 8 songs with 4 Chinese songs and 4 English songs. We use SongEcho [11] and ACE-Step 1.5 [12] as baselines. Suno-v5.5 is excluded since generating cover versions from human recordings is not supported.

The results are shown in Table 15. MuLaCover consistently outperforms baselines on most metrics, which suggests that our approach can generalize to unseen human-composed tracks despite primarily trained on synthetic data. We note that this evaluation is small in scale, and leave a more comprehensive study on real-world data to future work.

H Preservation of Basic Text-to-Music Ability

MuLaCover incorporates classifier-free guidance for lead-sheet conditioning. Therefore, to investigate whether the original text-to-music (T2M) capability is preserved, we replace the input lead-sheet condition with a null condition (denoted as MuLaCover-nc). Without the lead-sheet condition, MuLaCover-nc naturally falls back to a pure text-to-music generation setting. We evaluate MuLaCover-nc and the original HeartMuLa model on the same test set. Results are shown in Table 16.

Overall, MuLaCover-nc achieves comparable performance to the original HeartMuLa model across most metrics, indicating that the LoRA fine-tuning does not introduce significant catastrophic forgetting of the original T2M capability. Compared with HeartMuLa, MuLaCover-nc achieves slightly better performance on three out of four AudioBox metrics (PQ, CE, and CU), while showing moderate degradation on some SongEval dimensions (Coh., Mem., Nat., Cla., and Mus.). Meanwhile, the lyric and style-following metrics (PER and MuQ-Sim) are improved after fine-tuning.

Table 16: Comparison between MuLaCover-nc and HeartMuLa [10]. ↑ / ↓ indicate higher / lower is better. **Bold** marks the better result in each column.

	AudioBox [54] (↑)				SongEval [55] (↑)					Lyrics	Style	
Method	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)
HeartMuLa [10]	8.23	**6.32**	7.61	7.89	**4.68**	**4.69**	**4.51**	**4.58**	**4.60**	0.15	0.27	**0.59**
MuLaCover-nc	**8.30**	6.18	**7.68**	**7.92**	4.60	4.59	4.34	4.43	4.47	**0.11**	**0.41**	**0.59**

Table 17: Robustness of MuLaCover to lead-sheet extraction errors on Task 1. p_1 and p_2 denote the probabilities of note removal and pitch shifting, respectively. ↑ / ↓ indicate higher / lower is better.

Corruption		AudioBox (↑)				SongEval (↑)					Lyrics	Style		CSI judge	
p_1	p_2	PQ	PC	CE	CU	Coh.	Mem.	Nat.	Cla.	Mus.	PER (↓)	MuQ-Sim (↑)	MuQ-Dist (↑)	CSI-R@1 (↑)	CSI-Rank (↓)
0	0	8.34	6.34	7.70	7.93	4.59	4.59	4.34	4.44	4.47	0.13	0.41	0.48	0.97	1.05
0	5%	8.29	6.26	7.67	7.90	4.54	4.52	4.25	4.35	4.40	0.13	0.40	0.56	0.63	2.85
0	10%	8.26	6.22	7.61	7.87	4.48	4.46	4.18	4.29	4.34	0.14	0.40	0.57	0.42	5.10
5%	0	8.26	6.20	7.61	7.89	4.51	4.50	4.23	4.32	4.37	0.13	0.39	0.56	0.62	2.91
10%	0	8.29	6.26	7.70	7.91	4.57	4.56	4.28	4.39	4.44	0.11	0.38	0.57	0.44	3.69
10%	10%	8.30	6.23	7.70	7.92	4.52	4.51	4.23	4.33	4.37	0.12	0.38	0.58	0.28	5.93

We hypothesize that this improvement may be related to our training objective in Stage 3. During this stage, the model learns to generate cover versions conditioned on the lyrics and style tags of the target cover song, while the lead-sheet condition is extracted from the original song. Since the textual conditions and lead-sheet condition correspond to different versions of the song, the model is encouraged to better disentangle and utilize textual style/lyric information rather than relying solely on the musical condition. This may improve its ability to follow textual conditions even in the absence of lead-sheet input.

I Robustness to Noisy Lead-sheets

We investigate two common types of lead-sheet extraction errors, namely missing notes and inaccurate notes, and provide a brief robustness analysis. To simulate missing notes, each note in the extracted lead sheet is randomly removed with probability p_1. To simulate inaccurate notes, we first sample a pitch shift (in semitones) uniformly from $[-3, 3]$ for each note, and then shift the note by this amount with probability p_2. We evaluate MuLaCover on Task 1 under different (p_1, p_2) settings, as reported in Table 17.

Overall, MuLaCover is robust to moderate lead-sheet extraction errors in terms of generation quality: both AudioBox and SongEval scores exhibit only limited degradation across all corruption levels, and lyric accuracy remains stable. In contrast, cover identification performance decreases as the corruption level increases. This is expected, since a less accurate lead sheet provides weaker guidance on the musical identity of the reference song. Interestingly, MuQ-Dist increases under corrupted conditions. We hypothesize that noisy conditions introduce additional deviations from the original song, encouraging more diverse reinterpretations and thus increasing the distributional distance between the generated cover and the reference.

J Lead-sheet Controllability

We additionally report lead sheet controllability for Task 1. We consider melody similarity (MS) and chord similarity (CS). Specifically, we extract the frame-level melody and chord sequences from the generated song, represent them as pitch/chroma vectors, and compute the frame-wise cosine similarity with the corresponding sequences of the input lead sheet, averaged over all frames. For models whose outputs are not frame-aligned with the input (MuLaCover and Suno-v5.5), we first align the two sequences with Dynamic Time Warping (DTW) and then compute the similarity along the optimal alignment path. See Table 18 for results.

For lead-sheet control, MuLaCover follows the input chord progression most closely of all methods. SongEcho attains higher melody similarity because it conditions on frame-wise F0 from the source vocals, which closely matches the frame-level metric, whereas our tempo-normalized melody

Table 18: Lead sheet metrics on the full-song cover task (Task 1). ↑ indicates higher is better. **Bold** marks the best and underline the second-best result in each column.

	Lead sheet	
Method	MS (↑)	CS (↑)
SongEcho [11]	**0.77**	0.74
ACE-Step-1.5 [12]	0.63	<u>0.86</u>
Suno-v5.5 [53]	0.57	0.82
MuLaCover	<u>0.68</u>	**0.89**

condition allows rhythmic flexibility at the cost of a slight alignment error. Among methods without frame-level alignment, MuLaCover still follows the melody more closely than Suno-v5.5.

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: The three contributions enumerated at the end of the abstract and the Introduction—the BPM-normalized symbolic lead-sheet formulation with gated cross-attention injection, the cross-cover finetuning curriculum together with a curated paired dataset, and the combined objective/listening/expert-interview evaluation—map one-to-one to the methods in Section 3 and the results in Section 4.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: Section 5 discusses the limitations of our formulation: the invariant/variant boundary of a cover is inherently open-ended, whereas MuLaCover fixes it at the lead-sheet level and does not yet let users adjust how closely the output follows the source. Appendix I further quantifies the sensitivity of cover identifiability to lead-sheet extraction errors, and Appendix G notes that our evaluation on human recordings is small in scale (8 songs).

   Guidelines:

   - The answer [N/A] 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: [N/A]

   Justification: The paper is a system/empirical contribution and does not state any formal theorems or proofs.

   Guidelines:

   - The answer [N/A] 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: The architecture is fully specified in Section 3 (including Eqs. 1–5); the data construction, three-stage curriculum and hardware are described in Section 4.1; the Stage 3 cross-cover pipeline is detailed in Appendix A; per-stage hyperparameters appear in Appendix B (Table 5); and task setups and baseline configurations are in Appendix C.

   Guidelines:

   - The answer [N/A] 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: Audio samples and source code are available at our project page as stated in the abstract. The construction pipeline of the cross-cover paired dataset is specified in Appendix A, and the training configuration needed to reproduce the main results is given in Section 4.1 and Appendix B.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Data pipeline, backbone choice, inserted-layer count $L{=}8$, and Stage 3 LoRA setup appear in Section 4.1; objective test-set construction, metrics, and CSI/MuQ-MuLan/AudioBox/SongEval evaluators are listed in Section 4.3; per-stage learning rates, epochs, batch sizes, optimizer (AdamW), scheduler, and the dynamic mask schedule are tabulated in Appendix B (Table 5); baseline-specific inference settings are documented in Appendix C.

   Guidelines:

   - The answer [N/A] 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: Table 3 in Section 4.4 reports per-metric means with 95% confidence intervals over the (participant, song) pairs, and Appendix E reports the Friedman omnibus test (Table 11) and Bonferroni-corrected pairwise Wilcoxon signed-rank tests on the within-block score differences (Table 12), with the paired-block analysis unit explicitly defined in the text.

   Guidelines:

   - The answer [N/A] 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: Section 4.1 states that all experiments run on 16×NVIDIA A100 (80 GB) GPUs, and Table 5 in Appendix B lists the per-stage epoch wall-clock time and trainable-parameter counts for each of the three stages.

   Guidelines:

   - The answer [N/A] 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: We have reviewed the NeurIPS Code of Ethics. The data sources (Section 4.1 and Appendix A), the human-subjects protocol (Appendix E and F), and the copyright and responsible-use statement in Section 5 comply with it.

Guidelines:

- The answer [N/A] 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: Positive impacts are discussed in Sections 2.2 and 5: a lead-sheet-conditioned cover generator provides an editable, interpretable control interface that can be integrated into compositional pipelines to support human–AI music co-creation. Negative impacts are discussed in Section 5: generated covers may constitute derivative works of existing songs, raising copyright concerns; we state that users are responsible for obtaining permissions from rights holders before distributing or commercially using such outputs.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [Yes]

Justification: All training data are synthetic and obtained from music generation providers in accordance with their terms of service, so no copyrighted human recordings are used in training (Section 5). Section 5 further requires users to obtain the necessary permissions from rights holders before distributing or commercially using generated covers.

Guidelines:

- The answer [N/A] 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: All external models, baselines, and tools—HeartMuLa, SongEcho, ACE-Step-1.5, Suno v5.5, Demucs, Whisper, All-in-One, SongFormer, Qwen2-Audio, DeepSeek, AudioBox-Aesthetics, SongEval, MuQ-MuLan, CoverHunter, and the LoRA / RoPE implementations—are cited with their original publications and official repositories in Section 4.1, Section 4.3, and Appendix C, and we use them in accordance with their respective open-source licenses and API terms of service.

    Guidelines:

    - The answer [N/A] 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 two new assets are the MuLaCover model and the cross-cover paired dataset. The model architecture, training curriculum, and hyperparameters are documented in Section 3, Section 4.1, and Appendix B. The dataset construction is documented in Appendix A. Documentation and license accompany the release on the project page.

    Guidelines:

    - The answer [N/A] 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: The full questionnaire interface (welcome page, demographics page, and per-case rating page with verbatim metric anchors) is shown in Figure 2 of Appendix E; the expert-interview protocol and the LLM summarization prompt are reproduced in Appendix F. All participants in the listening study and the expert interview were unpaid volunteers, as stated at the end of Appendix E.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: As described in Appendix E, the listening study only collects anonymous audio preference ratings and basic demographics from unpaid volunteers who give informed consent before starting; the expert interview (Appendix F) likewise involves only verbal commentary on generated audio.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLM usage is disclosed where it occurs: Section 4.1 and Appendix A declare that DeepSeek and Qwen2-Audio are used to generate / format the style-tag vocabulary and to synthesize the test-set lyrics and Stage 3 paired data; Appendix F reproduces the verbatim prompt template used to convert the organized expert record into the summary verdict quoted in Section 4.4. The LLMs are not part of the core modeling of MuLaCover.

Guidelines:

- The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.