LeVo: High-Quality Song Generation with Multi-Preference Alignment

Shun Lei[1†], **Yaoxun Xu**[1†], **Zhiwei Lin**[1], **Huaicheng Zhang**[3], **Wei Tan**[2*], **Hangting Chen**[2*], **Yixuan Zhang**[2], **Chenyu Yang**[4], **Haina Zhu**[5], **Shuai Wang**[6], **Zhiyong Wu**[1*], **Dong Yu**[2*]
[1] Shenzhen International Graduate School, Tsinghua University, Shenzhen
[2] Tencent AI Lab [3] Wuhan University
[4] The Chinese University of Hong Kong, Shenzhen (CUHK-Shenzhen), Shenzhen, China
[5] X-LANCE Lab, Shanghai Jiao Tong University, Shanghai
[6] School of Intelligence Science and Technology, Nanjing University, Suzhou, China
`{leis21, xuyx22}@mails.tsinghua.edu.cn, erichtchen@tencent.com, zywu@sz.tsinghua.edu.cn`

Abstract

Recent advances in large language models (LLMs) and audio language models have significantly improved music generation, particularly in lyrics-to-song generation. However, existing approaches still struggle with the complex composition of songs and the scarcity of high-quality data, leading to limitations in audio quality, musicality, instruction following, and vocal-instrument harmony. To address these challenges, we introduce LeVo, a language model based framework consisting of LeLM and Music Codec. LeLM is capable of parallel modeling of two types of tokens: mixed tokens, which represent the combined audio of vocals and accompaniment to achieve better vocal-instrument harmony, and dual-track tokens, which separately encode vocals and accompaniment for high-quality song generation. It employs two decoder-only transformers and a modular extension training strategy to prevent interference between different token types. To further enhance musicality and instruction following ability, we introduce a multi-preference alignment method based on Direct Preference Optimization (DPO). This method handles diverse human preferences through a semi-automatic data construction process and post-training. Experimental results demonstrate that LeVo significantly outperforms existing open-source methods in both objective and subjective metrics, while performing competitively with industry systems. Ablation studies further justify the effectiveness of our designs. Audio examples and source code are available at https://levo-demo.github.io/ and https://github.com/tencent-ailab/songgeneration.

1 Introduction

Songs represent one of the most significant forms of musical expression, encapsulating human creativity and intelligence. By blending expressive vocals with a rich tapestry of instruments, songs convey the emotions and thoughts of their creators, offering artistic appeal and broad cultural influence. Rapid advances in Artificial Intelligence Generated Content (AIGC) have already transformed creative domains—from text generation [1–3] and speech synthesis [4–7] to more sophisticated domains such as image generation [8–10] and instrumental music generation [11–14]. Nevertheless, creating songs remains a complex challenge for AIGC systems: they must generate high-quality vocal and instrumental tracks and integrate them seamlessly while maintaining musicality and

[*]Corresponding author, [†] Equal contribution.

following instructions. These challenges are amplified for long-form song generation, where modeling complexity and computational cost increase significantly.

Early song generation approaches treat the combined audio of the vocals and accompaniment as a single prediction target. Jukebox [15] introduced the idea of using language models (LMs) to predict discrete codes ("mixed tokens") extracted from the song audio. Later studies, such as SongCreator [16] and MusiCot [17], have further optimized the language model to improve musicality. Despite their success, mixed token-based approaches exhibit a critical limitation: the restricted vocabulary cannot fully capture the intricate combination of vocals and accompaniment, which leads to lower sound quality. YuE [18] tackles this limitation by introducing a dual-track token prediction strategy that generates separate token sequences for the vocal and instrumental tracks. SongGen [19] extends this idea and shows that interleaving the two streams along the temporal dimension, instead of predicting them in parallel, reduces interference between token types and achieves better performance. While these methods produce high-quality vocals and accompaniments, they struggle to maintain vocal-instrument harmony due to the independent prediction. The interleaved prediction mode dramatically increases the length of the token sequence, which limits scalability and makes it difficult to maintain musicality and follow instructions.

Moreover, available datasets in the song generation community have long been constrained by highly uneven quality and unreliable music annotations, which exhibit several challenges. On the one hand, models trained on such noisy data are generic enough to model song signals, but they lack prior knowledge of musicality to bias the generated songs toward listener-preferred music. On the other hand, unreliable annotations limit the model's ability to follow instructions such as lyrics and prompts. Similar issues in text generation are typically circumvented by using the reinforcement learning (RL) method (e.g., RLHF), but music annotation requires specialized domain knowledge and evaluation along multiple perceptual dimensions.

This paper presents LeVo, a language model based song generation framework consisting of LeLM and Music Codec. LeLM is used to model mixed tokens and dual-track tokens in parallel. Mixed tokens guide the overall arrangement of melody, rhythm, and tempo, ensuring vocal-instrument harmony, while dual-track tokens capture finer nuances to improve the sound quality and musicality. The Music Codec then reconstructs the dual-track tokens into high-fidelity music audio. To prevent interference between these two token types, the proposed LeLM utilizes a language model with an autoregressive (AR) decoder and a modular extension training strategy. This framework allows flexible control over musical elements, enabling text-based descriptions to influence instrumental, genre, and emotion, and audio prompts to achieve zero-shot style transfer. Additionally, to address the challenges of musicality and instruction following, we further introduce a DPO-based [20] multi-preference alignment method to improve the model's performance across multiple dimensions. The main contributions of this paper are summarized as follows:

- LeVo presents a new paradigm to optimize LM-based music generation by three-stage training: pre-training, modular extension training, and multi-preference alignment. Pre-training brings generation diversity and vocal-instrument harmony, modular extension training improves sound quality and musicality without disrupting the pre-trained knowledge, and multi-preference alignment further enhances instruction following and musicality.

- The architecture of LeLM is carefully designed using a language model with an AR decoder to predict mixed and dual-track tokens in parallel. Compared to single token prediction (only using mixed or dual-track tokens) or straightforward joint training, our method simultaneously optimizes musicality, vocal–instrument harmony, and sound quality.

- To balance the multi-dimensional demands of song generation, we propose a multi-preference alignment method based on DPO, which is the first attempt to apply multi-preference DPO to song generation. Our strategy handles diverse human preferences by corresponding semi-automatic data construction processes and DPO fine-tuning. The final combined model achieves higher musicality and better instruction following.

In terms of musicality, sound quality, vocal–instrument harmony, and instruction following, the proposed LeVo model significantly improves over the open-source music generation models and performs competitively with current state-of-the-art industry systems.

2 Related Work

Music Generation Early efforts [21, 22] in music generation primarily focused on symbolic music, which is limited by fixed instruments and lacks expressiveness. Recently, LLMs have demonstrated strong reasoning capabilities and scaling behaviors [1–3]. Several works [11, 14, 23] have achieved the end-to-end music generation using language models. These works encode music as discrete token sequences by the Vector Quantised-Variational Autoencoder (VQ-VAE) [24] or the Residual Vector Quantization (RVQ) [25, 26], and then process them with language models. However, their performance was constrained by quantization loss. To address this, some studies adopt diffusion models, which have been proven effective at modeling continuous representations [27, 28]. However, these methods face challenges in long-context generation tasks due to the significantly increased computational demands as the sequence lengthens and the model size grows. Recently, MeLoDy [12] and AudioLDM 2 [29] combined language models and diffusion models, achieving state-of-the-art (SOTA) performances with high-fidelity and musicality. We extend this idea by using a diffusion model as the decoder of the codec and LeLM, which considers the complex components of songs, to better achieve song generation.

Furthermore, several intriguing studies have focused on applying RL in music generation. BATON [30] integrates a reward model, which is trained based on human preferences, into the standard diffusion loss to guide the training of diffusion models. MusicRL [31] fine-tunes MusicLM [14] using reward functions derived from large-scale user feedback through Reinforcement Learning from Human Feedback (RLHF) to learn human preferences. Tango2 [32] semi-automatically generates preference datasets for DPO [20] training, providing a cheaper and more robust alternative. Different from these works that only focus on one specific dimension in non-vocal music generation, to our knowledge, we are the first attempt to align multiple preferences in song generation.

Song Generation Recently, several studies have explored the challenging task of song generation, which involves generating vocals with rich accompaniment. Jukebox [15] is one of the pioneering efforts, which utilizes a multi-scale VQ-VAE to compress songs into discrete codes and models them with a cascaded transformer model. To improve the musicality of generated songs, SongCreator [16] introduced a dual-sequence language model to capture the relationship between vocals and accompaniment. MusiCot [17] predicts coarse-grained style representations to first sketch an overall musical structure before generating audio tokens, thereby improving the coherence and creativity of the generated songs. However, these approaches are affected by limited vocabulary and interference between vocals and accompaniment. To address these issues, Melodist [33] and MelodyLM [34] utilize a multi-stage process to sequentially produce vocals and accompaniment. YUE [18] and SongGen [19] attempt to operate on sequences consisting of vocal tokens and accompaniment tokens. Different from these, DiffRhythm [35] uses a diffusion-based approach to generate full-length songs. Despite the advancements in these methods, they still struggle with the complex composition of songs and the uneven quality of data, which makes it difficult to maintain musicality, lyrics alignment, and vocal-instrument harmony. Industry tools such as Meruka [36], Seed-Music [37], Suno [38], and Udio [39] have demonstrated promising results for song generation, but none have fully disclosed their technical details. Our proposed LeVo outperforms previous academic methods by predicting mixed and dual-track tokens in parallel while explicitly preventing their mutual interference. Furthermore, by leveraging a DPO-based multi-preference alignment method, LeVo achieves additional improvements, yielding human-perceptual quality that is competitive with leading industry tools.

3 Method

3.1 Overview

Let $\mathbf{x} \in \mathcal{X}$ represent a song audio. A song generation process can be defined as $f : \mathcal{C} \mapsto \mathcal{X}$, where $\mathcal{C}$ is the set of conditions. However, end-to-end generating a high-fidelity song $\mathbf{x}$ from $\mathbf{C}$ with a neural network f remains challenging to date. In the same spirit as previous works [16, 18], we introduce a language-alike token sequence, denoted as $\mathbf{S} = (S_1, \ldots, S_N)$, to capture significant structural information within the song and to embody language models as the "brain" for creating songs.

Our proposed LeVo generates songs from lyrics and can be supplemented with text descriptions and audio prompts. As illustrated in Figure 1, it consists of LeLM and Music Codec. The encoder

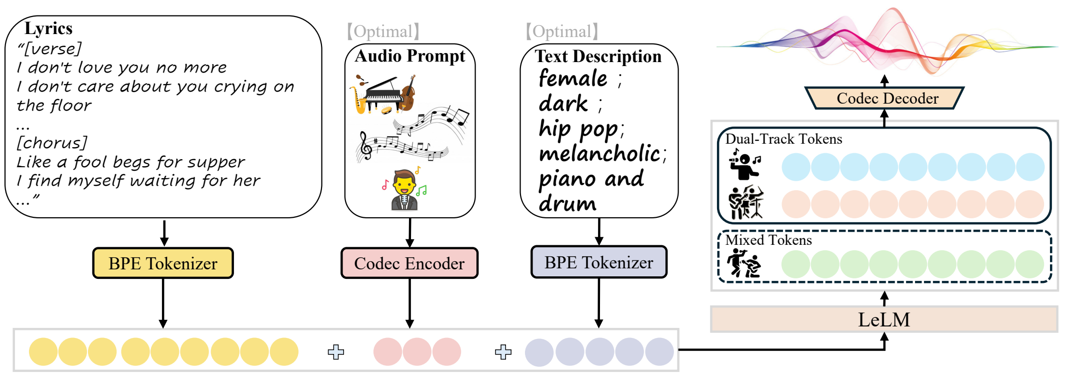


Figure 1: The overview of LeVo, a song generation framework based on lyrics, optional text descriptions, and optional audio prompts. It consists of LeLM and Music Codec.

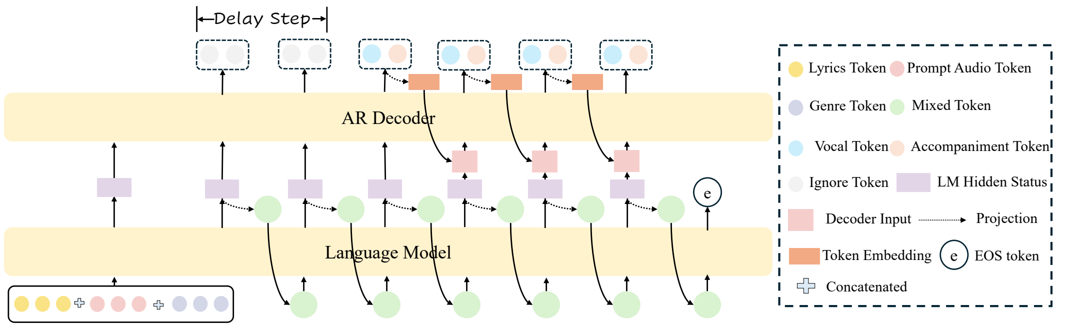


Figure 2: The architecture of LeLM, which consists of a language model and an AR decoder.

of Music Codec is utilized to extract mixed tokens (i.e., $\mathbf{S}_m \in \mathcal{S}_m$), vocal tokens (i.e., $\mathbf{S}_v \in \mathcal{S}_v$), and accompaniment tokens (i.e., $\mathbf{S}_a \in \mathcal{S}_a$) from the song audio as prediction targets for LeLM. To parallelly predict the mixed tokens and dual-track tokens given $\mathbf{C}$, we propose a language model with an AR decoder named LeLM, as illustrated in Figure 2. The lyrics, optional text descriptions, and optional audio prompts are concatenated and fed into the LeLM as prefix context. Then, the decoder of Music Codec is used to generate high-quality and high-fidelity song audio based on dual-track tokens that contain more detailed information about the vocal and accompaniment tracks.

Furthermore, to address the challenges posed by interference between different types of tokens and issues such as musicality and instruction following, we propose a three-stage training strategy and a DPO-based multi-preference alignment method. In the remainder of this section, we present the LeVo architecture details and the training process described above.

3.2 Language Modeling of LeVo

Although modeling dual-track tokens struggles to maintain vocal-instrument harmony, its significant advantages in sound quality have led to widespread attempts [18, 19]. The interleaving pattern, in particular, has shown improved performance over the parallel pattern by reducing interference between different token types. However, limited by the substantial increase in sequence length, the interleaving pattern struggles to scale to long-context song generation. To address this, LeVo introduces a novel parallel modeling approach that avoids cross-token interference without significantly lengthening the sequence, thereby improving both vocal-instrument harmony and sound quality.

As shown in Figure 2, this approach consists of a language model and an autoregressive (AR) decoder. The language model utilizes a decoder-only Transformer architecture, which has been widely adopted in previous works on music generation [12, 14, 18]. It focuses on the next-token prediction task for mixed tokens to capture the high-level structural information about the song, such as melody, rhythm,

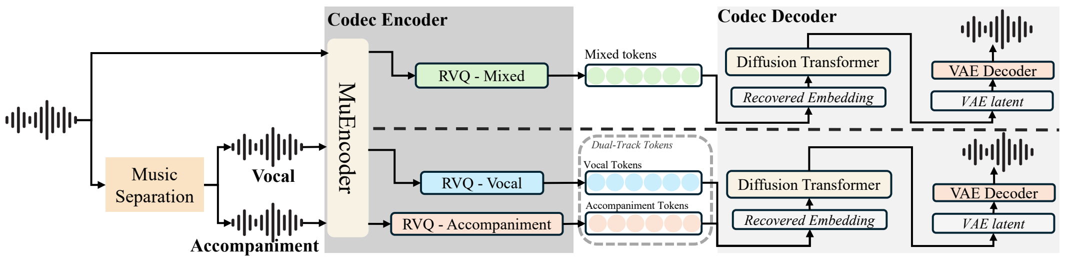


Figure 3: The framework of the Music Codec in LeVo.

and tempo, ensuring vocal-instrument harmony. This task can be formulated as:

$$p(\mathbf{S}_m|\mathbf{C};\boldsymbol{\theta}) = \prod_{t=0}^{T} p(\mathbf{S}_{m,t}|\mathbf{S}_{m,<t},\mathbf{C};\boldsymbol{\theta}) \tag{1}$$

where $\mathbf{C}$ denotes conditions, including lyrics, optional text descriptions, and optional audio prompts.

To facilitate high-quality song generation, we use an AR decoder that predicts dual-track tokens in parallel based on the outputs of the language model. It also utilizes a decoder-only Transformer architecture with significantly fewer layers and parameters than the language model and is designed to refine the information generated by the language model through modeling of finer acoustic details. As the hidden states of the language model contain richer semantic and acoustic information than the mixed tokens, we pass these hidden states to the AR decoder. Specifically, at each time step, the embeddings of the predicted vocal and accompaniment tokens from the previous time step, along with the hidden states of the language model's last layer, are concatenated to form the input of the AR decoder. Two groups of linear heads are then employed to predict the audio tokens for each track based on the output of the AR decoder. Moreover, to provide rich contextual information, we introduce a delay pattern between the dual-track tokens and mixed tokens. This means that when predicting the dual-track tokens at the t-th time step, the model can consider not only the language model's output at time steps less than or equal to t, but also the output for the next k time steps, where k represents the number of time steps for the delay. This process can be simply represented as:

$$\begin{aligned} p(\mathbf{S}_v,\mathbf{S}_a|\mathbf{C};\boldsymbol{\theta}) = \prod_{t=0}^{T-k} & p(\mathbf{S}_{v,t},\mathbf{S}_{a,t}|\mathbf{S}_{v,<t},\mathbf{S}_{a,<t},\mathbf{S}_{m,<t+k},\mathbf{C};\boldsymbol{\theta}) \\ \prod_{t=T-k+1}^{T} & p(\mathbf{S}_{v,t},\mathbf{S}_{a,t}|\mathbf{S}_{v,<t},\mathbf{S}_{a,<t},\mathbf{S}_{m},\mathbf{C};\boldsymbol{\theta}) \end{aligned} \tag{2}$$

3.3 Music Codec in LeVo

Building upon MuCodec [40]—an efficient 48kHz music codec capable of operating at low bitrates—we have developed the Music Codec in LeVo, as shown in Figure 3. The codec consists of an encoder and a decoder. The encoder is composed of the MuEncoder [40] and an RVQ: the MuEncoder extracts music-relevant representations, while the RVQ discretizes these representations into tokens. As the central intermediary in LeVo, these tokens serve not only as the prediction targets for LeLM but also play a pivotal role in achieving high-fidelity music reconstruction by acting as inputs for the decoder. The codec decoder comprises a diffusion transformer and a Variational Autoencoder (VAE) decoder. The diffusion transformer reconstructs VAE features from the token-derived embeddings, which the VAE decoder then converts directly into audio. This approach is significantly faster than methods that rely on Mel-spectrograms as intermediate steps for audio reconstruction.

Music inherently exhibits a rich hierarchical structure, particularly in the complex interplay between vocals and accompaniment. To generate expressive vocals alongside clear and diverse accompaniment, we propose two strategies: the mixed token approach and the dual-track token approach. In the mixed token approach, music is processed by the encoder to extract mixed tokens. These tokens encapsulate information from both the vocals and the accompaniment. The decoder then uses the recovered embeddings from these mixed tokens to reconstruct the VAE latent, ultimately restoring the original music. In the dual-track token approach, a pretrained music separation model is employed to separate the vocal and accompaniment tracks. Each track is then processed by its respective encoder to produce vocal tokens and accompaniment tokens. The decoder conditions on the recovered embeddings from both sets of tokens to reconstruct the VAE latent of the original music, and finally restores the music.

3.4 DPO-based Multi-Preference Alignment

The scarcity of high-quality data has severely constrained the effectiveness of previous works in academic research. On the one hand, uneven data quality prevents models from learning prior knowledge about musicality. On the other hand, the lack of accurate music annotations, such as lyrics and descriptive text, undermines the model's ability to perform lyrics alignment and prompt-driven control. To address these challenges, we first adopt a semi-automatic multi-dimensional preference data construction method to generate large-scale preference data at a low cost for DPO fine-tuning.

First, we generate 20,000 lyrics by using a large language model. For each lyric, we randomly select an audio prompt from a pre-defined dataset that includes text descriptions. Then, we generate multiple samples under three different conditions: using only the text description, using only the audio prompt, and using both the text description and the audio prompt. For the three dimensions of lyrics alignment, prompt consistency, and musicality, we then introduce three distinct strategies for constructing preference data corresponding to each dimension.

Strategy 1: Lyric Alignment Preferences The lyrics of the generated audio might exhibit misalignment with the input text, for example, word substitution, insertion, and deletion. To eliminate such a misalignment, we construct win-lose pairs using phoneme errors calculated by automatic speech recognition (ASR). Two audio samples will form a win-lose pair if their phoneme error numbers are larger than 40. We pick out all pairs satisfying the phoneme error gap condition.

Strategy 2: Prompt Consistency Preferences Considering the noise observed in the annotation of the training data, the style of the generated audio might be inconsistent with the input prompt. To improve the ability to prompt-driven control, we construct preference pairs using similarity scores calculated by the MuQ-MuLan [41] model. We aim to ensure that the winning audio sample is both strongly aligned with the prompt and substantially better aligned than the losing sample. For text descriptions, the winning sample must achieve a score of at least 0.3, exceeding the losing sample's score by at least 0.1. For audio prompts, the winning sample must reach a score of at least 0.75, again with a score difference greater than 0.1 compared to the losing sample.

Strategy 3: Musicality Preferences To enhance musicality, we employ a three-stage procedure to sift through win-lose pairs. Initially, we carry out crowdsourcing to rank a subset of the generated samples, which included 4,000 different sets of lyrics. We select those with high levels of agreement among human evaluators, forming a dataset of win-lose pairs. In the second stage, we train a reward model based on the datasets curated from the first stage. Finally, in the third stage, we deploy the reward model across all generated samples. We establish a threshold to filter out data that exhibits a large gap in reward scores and tune the threshold to ensure that the selected win-lose pairs have an accuracy above 80%. Through this three-stage procedure, we ultimately collect roughly 60,000 win–lose pairs. The rationale behind this procedure is that while the size of the human-labeled dataset is limited, the use of a reward model enables us to identify all potential pairs.

The above paired data are then used for DPO fine-tuning to address the challenges posed by scarce high-quality data in the dimensions of lyrics alignment, prompt consistency, and musicality. To balance different aspects of music generation, we introduce an interpolation-based multi-preference alignment method inspired by DNI (Deep Network Interpolation) [42]. In this mode, we fine-tune the model separately on the preference data curated for each strategy to obtain the network parameters dedicated to each preference. Then, we apply linear interpolation across all parameters of these three specialized networks to produce the final model. Moreover, this approach also supports smooth transitions in performance according to specific demands by the controllable interpolation coefficient.

3.5 Three-Stage Training Paradigm

To facilitate the performance of LM-based song generation approaches, we introduce a novel three-stage training paradigm that progressively enhances the capabilities of LeLM:

Table 1: Objective results of comparison and ablation systems for song generation. The asterisk (*) denotes that we reproduce SongGen using our training data. The overall first and second results are marked with **bold** and underline, respectively.

Models	FAD ↓	MuQ-T ↑	MuQ-A ↑	PER ↓	Content Scores ↑			
					CE	CU	PC	PQ
Suno-V4.5	<u>2.59</u>	**0.34**	<u>0.84</u>	21.6	7.65	7.86	5.94	8.35
Haimian	2.97	0.22	–	11.8	7.56	7.85	5.89	8.27
Mureka-O1	**2.50**	0.33	**0.87**	**7.2**	7.71	7.83	**6.39**	<u>8.44</u>
YuE	2.65	0.27	0.74	36.4	7.13	7.39	5.90	7.77
DiffRhythm	4.86	0.26	0.51	12.3	6.65	7.32	5.71	7.77
ACE-Step	2.69	0.28	–	37.1	7.37	7.52	<u>6.26</u>	7.85
SongGen*	2.68	0.25	0.80	27.5	7.63	7.79	5.94	8.37
LeVo	2.68	**0.34**	0.83	**7.2**	**7.78**	**7.90**	6.03	**8.46**
w/o Train stage 2	2.71	0.28	0.82	17.5	<u>7.76</u>	7.81	5.69	8.44
w/o AR decoder	2.83	0.27	0.80	26.0	7.54	7.71	5.61	8.32
w/o Dual-track	2.83	0.33	0.83	11.0	7.72	<u>7.88</u>	5.82	8.43
w/o DPO	2.60	0.31	0.82	10.6	7.70	7.86	5.89	8.39

Stage 1: Pre-training The language model in LeLM is first trained on large-scale music data to align the modalities between the various conditioning inputs and the mixed tokens. To ensure the model functions under different input conditions, both audio prompts and text descriptions undergo 50% random drop. At this stage, the AR decoder is frozen to facilitate the language model to focus solely on mixed tokens, bringing generation diversity and vocal-instrument harmony.

Stage 2: Modular Extension Training Building on Stage 1, we further train the AR decoder in LeLM to model dual-track tokens in parallel. This captures fine-grained acoustic details and nuances, thereby improving sound quality and musicality. To preserve the pre-trained knowledge, all modules trained in Stage 1 are frozen.

Stage 3: Multi-preference alignment Finally, we employ the semi-automatic data construction processes to create a multi-dimensional preference dataset and fine-tune the entire LeLM using the DPO loss. This stage significantly enhances musicality and the ability to follow instructions.

4 Experiments

4.1 Experimental setup

Dataset LeVo is trained on a large-scale music dataset comprising 2 million songs (approximately 110,000 hours). We adopt an automatic data processing pipeline similar to AutoPrep [43] to address the challenge of the lack of accurate annotations. Initially, we use the Demucs [44, 45] to extract the vocals from the songs. Whisper [46] and wav2vec 2.0 [47, 48] are then leveraged for lyric recognition and to provide timestamps for the lyrics. To enable the model to learn the structure of the songs, we adopt the All-In-One model [49] to automatically extract music segments (e.g., verse, chorus, etc.). Additionally, we annotate all tracks utilizing Qwen2-Audio [50] to obtain open-vocabulary tags.

Model setup Our proposed LeLM has approximately 2B parameters. For the MuEncoder model that generates mixed tokens and dual-track tokens, we utilize the pre-trained weights from Mucodec [40] with 300M parameters and a frame rate of 25HZ. Our diffusion model features a model size of about 700M parameters to convert tokens into high-quality waveforms. We replicate the VAE used in Stable Audio with 150M parameters. Additionally, for lyrics and text description, LeVo employs a byte pair encoding (BPE)-based Qwen2 tokenizer [51] to process raw text. Detailed configurations and training setups can be found in the Appendix B.

Evaluations For objective evaluation, both the Fréchet Audio Distance (FAD) [52] and the Phoneme Error Rate (PER) are employed. To calculate PER, the vocal track is first extracted by Demucs, and then Whisperlarge-v2 is utilized for lyric recognition. We employ MuQ-MuLan [41], a contrastive

Table 2: Subjective results of comparison and ablation systems for song generation. The asterisk (*) denotes that we reproduce SongGen using our training data. The overall first and second results are marked with **bold** and <u>underline</u>, respectively.

Models	MOS ↑					
	OVL	**MEL**	**HAM**	**SSC**	**AQ**	**LYC**
Suno-V4.5	**3.59**	**4.10**	**3.93**	**4.19**	**4.00**	3.17
Haimian	3.05	3.51	3.55	3.62	3.87	<u>3.32</u>
Mureka-O1	<u>3.42</u>	3.88	3.89	<u>4.14</u>	3.87	<u>3.32</u>
YuE	2.45	3.04	2.94	3.53	3.08	2.41
DiffRhythm	2.60	3.18	3.22	3.55	3.09	2.69
ACE-Step	2.26	3.02	3.30	3.21	2.36	2.22
SongGen*	2.91	3.43	3.44	3.66	3.69	2.84
LeVo	<u>3.42</u>	<u>3.93</u>	<u>3.90</u>	4.09	<u>3.96</u>	**3.38**
w/o Train stage 2	3.29	3.76	3.77	3.80	<u>3.96</u>	2.91
w/o AR decoder	2.93	3.44	3.34	3.59	3.71	2.74
w/o Dual-track	3.25	3.82	3.84	3.96	3.86	3.18
w/o DPO	3.18	3.71	3.76	3.97	3.93	3.18

music-language pre-training model, to measure the similarity between the generated song and its text description (MuQ-T) or audio prompt (MuQ-A). Additionally, the Meta Audiobox-Aesthetic [53] is used to capture perceived musical aesthetics, including content enjoyment (CE), content usefulness (CU), production complexity (PC), and production quality (PQ). For subjective evaluation, we conduct a mean opinion score (MOS) listening test. Specifically, we enlist ten music professionals to rate each generated song on a scale from 1 to 5 across six aspects: overall quality (OVL), focusing on musicality and naturalness; vocal melodic attractiveness (MEL); vocal-instrument harmony (HAM); song structure clarity (SSC); audio sound quality (AQ); lyrics following accuracy (LYC). None of the evaluators participated in model training to ensure objectivity. During the experiments, we asked participants to listen to the entire music generated by different models with the same input before scoring, thereby obtaining scores that reflect the relative differences between models. The Appendix D shows details.

Comparison systems We conducted a comprehensive comparison between LeVo and multiple systems. We selected three leading industry systems for benchmarking: Suno V4.5 [38], Mureka-O1 [36], and Haimian [54]. It is important to note that due to the black-box nature of these closed-source models, our evaluation conducted in May 2025 reflects the performance of these systems at that specific time. Furthermore, we selected four open-source academic systems for benchmarking: YuE[2] [18], DiffRhythm[3] [35], ACE-Step[4] [55] and SongGen [19]. Notably, the official SongGen model can only generate songs up to 30 seconds with a sampling rate of 16 kHz. For fair comparison, we reproduced its Interleaving (A-V) mode based on our dataset and codec.

4.2 Comparison with the State-of-The-Art Systems

Objective results Table 1 summarizes the objective results. LeVo achieves either superior or competitive performance across all metrics. It obtains the highest MuQ-T scores (0.34) and lowers PER (7.2%) among all systems, and the highest MuQ-A scores (0.83) after closed-source systems, which demonstrates its powerful instruction following ability, especially in text description and lyrics. LeVo also records the highest scores on the Audiobox-Aesthetic dimensions of CE (7.78), CU (7.90), and PQ (8.46), indicating its superior perceived musicality. For FAD and PC, LeVo achieves comparable results to other open-source approaches that focus on dual-track tokens modeling or diffusion-based generation, reflecting comparable sound quality, whereas industry model Mureka-O1 records the lowest FAD and higher PC, attributed to its exceptional sound quality.

[2]YuE is tested using the model released at https://github.com/multimodal-art-projection/YuE

[3]DiffRhythm is tested using DiffRhythm-full released at https://github.com/ASLP-lab/DiffRhythm

[4]ACE-Step is tested using the text2music demo space at https://huggingface.co/spaces/ACE-Step/ACE-Step

Table 3: Comparison of various DPO methods across multiple objective metrics. The overall first and second results are marked with **bold** and <u>underline</u>, respectively.

Models	FAD ↓	MuQ-T ↑	MuQ-A ↑	PER ↓	Content Scores ↑			
					CE	CU	PC	PQs
w/o DPO	**2.60**	0.31	0.82	10.6	7.70	7.86	5.89	8.39
with Strategy 1	2.85	0.30	0.81	**6.5**	7.72	7.86	5.97	8.42
with Strategy 2	2.89	**0.34**	**0.83**	10.3	7.75	7.87	5.96	8.43
with Strategy 3	<u>2.63</u>	0.32	0.82	11.2	**7.78**	**7.93**	**6.16**	<u>8.45</u>
Mixed Training	2.75	0.33	**0.83**	7.5	7.76	7.89	<u>6.04</u>	8.43
LeVo (Interpolation)	2.68	**0.34**	**0.83**	<u>7.2</u>	**7.78**	<u>7.90</u>	6.03	**8.46**

Subjective results As shown in Table 2, LeVo outperforms all open-source academic approaches on all dimensions, highlighting the musicality, audio quality, vocal-instrument harmony, and lyrics alignment of the generated song. It is important to note that our proposed parallel prediction method surpasses the interleaved pattern of SongGen, whose performance degrades on long-context song generation. Although the industry system Suno-V4.5 excels in most metrics, LeVo exceeds it by 0.21 points on LYC, which demonstrates LeVo's excellent lyric-alignment capability. Furthermore, LeVo achieves MOS scores close to Suno-V4.5 on OVL, MEL, HAM, and AQ, and outperforms all other systems, indicating its superior overall quality—especially in musicality, vocal-instrument harmony, and sound quality. However, Suno-V4.5 and Mureka-O1 show slightly better song structure clarity scores, which suggests room for improvement in music structural modeling.

4.3 Ablation study

Framework As presented in the lower part of Table 1 and 2, we demonstrate the effectiveness of several techniques used in LeVo, including the modular extension training strategy, the AR decoder, and the prediction of dual-track tokens. It is observed that jointly training the language model and AR decoder ("w/o train stage 2") decreases all metrics, particularly in PER, LYC, SSC, and MEL. Removing the AR decoder further amplifies this decrease. The results indicate that the proposed AR decoder and modular extension training strategy are crucial for preventing interference between mixed and dual-track tokens, thereby improving musicality and intelligibility while maintaining vocal-instrument harmony. Moreover, we find that removing the prediction of dual-track tokens causes a significant decline in FAD, PER, and PC (objective) together with OVL and LYC (subjective), whereas MEL, SSC, and AQ decline more modestly. It confirms that the prediction of dual-track tokens is essential for sound quality and intelligibility, and also has a few impact on musicality. We further explore the reconstruction performance of dual-track tokens in Appendix C.

DPO-based Multi-Preference Alignment As shown in the lower part of Table 1 and Table 2, removing DPO leads to significant decreases in PER, OVL, MEL, and LYC, indicating that DPO-based multi-preference alignment effectively enhances both musicality and lyric alignment. We also conducted a MOS listening test to evaluate the alignment between the given prompt and the generated music. In this experiment, "LeVo" achieved a score of 3.175, while "LeVo w/o DPO" scored 3.055, suggesting that DPO-based multi-preference alignment also improves prompt consistency.

Table 3 further details how different preferences affect objective performance. When the model is fine-tuned with a single preference, each strategy selectively enhances its targeted capability: Lyric Alignment Preference (Strategy 1) reduces PER from 10.6% to 7.0%; Prompt Consistency Preference (Strategy 2) yields the highest MuQ-T (0.34) and MuQ-A (0.83) scores; and Musicality Preference (Strategy 3) achieves the best Audiobox-Aesthetic results. We then compare our proposed interpolation-based multi-preference alignment method with a simple mixed training approach, where preference data from all strategies are combined during post-training. While mixed training improves both instruction following and musicality compared to the no-DPO baseline—albeit with a slight increase in FAD—our proposed interpolation method outperforms mixed training across most metrics, demonstrating a better balance among the multi-dimensional demands of music generation. We further analyze the smooth performance transitions enabled by the controllable interpolation coefficient in Appendix G.

5 Conclusion and Discussion

Conclusion In this paper, we present LeVo, a song generation model that parallelly predicts mixed tokens for vocal-instrument harmony and dual-track tokens for high-fidelity vocals and accompaniment. By leveraging the LeLM and its training strategy, LeVo eliminates the mutual influence between different types of tokens. In addition, we propose a multi-preference alignment method to improve the performance in terms of lyrics alignment, prompt consistency, and musicality. Experimental results demonstrate the excellent song generation capabilities of LeVo.

Limitations Despite LeVo outperforming all academic approaches, its audio quality remains constrained by the variable quality of training data and the use of discrete tokens, leaving a gap from state-of-the-art industry models. Moreover, there is still a disparity between LeVo and Suno across several subjective metrics, suggesting that further alignment with human preference is necessary. In addition, the scarcity and high cost of annotated music data have led us to rely heavily on pre-trained models for pseudo-labeling, which inevitably impacts LeVo's instruction following capability. On one hand, since text descriptions are generated using Qwen2-Audio, the diversity and richness of effective prompts are limited. On the other hand, errors accumulated in lyric extraction, structure recognition, and music captioning processes introduce noise into supervision, reducing the fidelity of instruction alignment. These factors collectively contribute to the remaining gap between LeVo and top-tier proprietary systems in both controllability and overall quality.

Broader Impact The proposed work has the potential to empower both content creators and novices to express their musical creativity with a low entry barrier, while also streamlining the workflow of experienced music producers. However, given LeVo's ability to perform style transfer and end-to-end song generation, it could be misused for creating misinformation, deepfake audio, or other harmful content. To promote responsible deployment, we plan to establish appropriate safeguards and usage guidelines when releasing the open-source code and model checkpoints. In addition, we examine the risks of training data memorization and content leakage, finding no evidence of direct reproduction from the training set under identical inputs (see Appendix H for detailed analysis).

Acknowledgements

This work is supported by National Natural Science Foundation of China (62076144) and Shenzhen Science and Technology Program (JCYJ20220818101014030).

References

[1] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[2] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[3] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[4] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *International Conference on Learning Representations*, 2021.

[5] Kai Shen, Zeqian Ju, Xu Tan, Eric Liu, Yichong Leng, Lei He, Tao Qin, sheng zhao, and Jiang Bian. Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers. In *The Twelfth International Conference on Learning Representations*, 2024.

[6] Ziyue Jiang, Jinglin Liu, Yi Ren, Jinzheng He, Zhenhui Ye, Shengpeng Ji, Qian Yang, Chen Zhang, Pengfei Wei, Chunfeng Wang, et al. Boosting prompting mechanisms for zero-shot speech synthesis. In *The Twelfth International Conference on Learning Representations*, 2023.

[7] Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, et al. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*, 2023.

[8] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[9] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. *Transactions on Machine Learning Research*, 2022.

[10] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved VQGAN. In *International Conference on Learning Representations*, 2022.

[11] 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, 2024.

[12] Max WY Lam, Qiao Tian, Tang Li, Zongyu Yin, Siyuan Feng, Ming Tu, Yuliang Ji, Rui Xia, Mingbo Ma, Xuchen Song, et al. Efficient neural music generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] Flavio Schneider, Zhijing Jin, and Bernhard Schölkopf. Moûsai: Text-to-music generation with long-context latent diffusion. *arXiv e-prints*, pages arXiv–2301, 2023.

[14] 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.

[15] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

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

[17] Max WY Lam, Yijin Xing, Weiya You, Jingcheng Wu, Zongyu Yin, Fuqiang Jiang, Hangyu Liu, Feng Liu, Xingda Li, Wei-Tsung Lu, et al. Analyzable chain-of-musical-thought prompting for high-fidelity music generation. *arXiv preprint arXiv:2503.19611*, 2025.

[18] 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.

[19] Zihan Liu, Shuangrui Ding, Zhixiong Zhang, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Dahua Lin, and Jiaqi Wang. Songgen: A single stage auto-regressive transformer for text-to-song generation. *arXiv preprint arXiv:2502.13128*, 2025.

[20] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

[21] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[22] Xipin Wei, Junhui Chen, Zirui Zheng, Li Guo, Lantian Li, and Dong Wang. A Multi-Scale Attentive Transformer for Multi-Instrument Symbolic Music Generation. In *Proc. INTERSPEECH 2023*, pages 5391–5395, 2023.

[23] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[24] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[25] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[26] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. High-fidelity audio compression with improved rvqgan. *Advances in Neural Information Processing Systems*, 36:27980–27993, 2023.

[27] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[28] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[29] Haohe Liu, Qiao Tian, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D Plumbley. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. *arXiv preprint arXiv:2308.05734*, 2023.

[30] Huan Liao, Haonan Han, Kai Yang, Tianjiao Du, Rui Yang, Qinmei Xu, Zunnan Xu, Jingquan Liu, Jiasheng Lu, and Xiu Li. Baton: aligning text-to-audio model using human preference feedback. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 4542–4550, 2024.

[31] Geoffrey Cideron, Sertan Girgin, Mauro Verzetti, Damien Vincent, Matej Kastelic, Zalán Borsos, Brian Mcwilliams, Victor Ungureanu, Olivier Bachem, Olivier Pietquin, et al. Musicrl: Aligning music generation to human preferences. In *International Conference on Machine Learning*, pages 8968–8984. PMLR, 2024.

[32] Navonil Majumder, Chia-Yu Hung, Deepanway Ghosal, Wei-Ning Hsu, Rada Mihalcea, and Soujanya Poria. Tango 2: Aligning diffusion-based text-to-audio generations through direct preference optimization. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 564–572, 2024.

[33] Zhiqing Hong, Rongjie Huang, Xize Cheng, Yongqi Wang, Ruiqi Li, Fuming You, Zhou Zhao, and Zhimeng Zhang. Text-to-song: Towards controllable music generation incorporating vocal and accompaniment. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6248–6261, 2024.

[34] Ruiqi Li, Zhiqing Hong, Yongqi Wang, Lichao Zhang, Rongjie Huang, Siqi Zheng, and Zhou Zhao. Accompanied singing voice synthesis with fully text-controlled melody. *arXiv preprint arXiv:2407.02049*, 2024.

[35] 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.

[36] Mureka team. Mureka-o1. *URL https://www.mureka.ai*, 2025.

[37] 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.

[38] Suno team. Suno v4.5. *URL https://suno.com/blog/introducing-v4-5*, 2025.

[39] Udio team. Udio v1.5. *URL https://www.udio.com/blog/introducing-v1-5*, 2024.

[40] Yaoxun Xu, Hangting Chen, Jianwei Yu, Wei Tan, Rongzhi Gu, Shun Lei, Zhiwei Lin, and Zhiyong Wu. Mucodec: Ultra low-bitrate music codec. *arXiv preprint arXiv:2409.13216*, 2024.

[41] 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. *arXiv preprint arXiv:2501.01108*, 2025.

[42] Xintao Wang, Ke Yu, Chao Dong, Xiaoou Tang, and Chen Change Loy. Deep network interpolation for continuous imagery effect transition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1692–1701, 2019.

[43] Jianwei Yu, Hangting Chen, Yanyao Bian, Xiang Li, Yi Luo, Jinchuan Tian, Mengyang Liu, Jiayi Jiang, and Shuai Wang. Autoprep: An automatic preprocessing framework for in-the-wild speech data. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1136–1140. IEEE, 2024.

[44] Alexandre Défossez. Hybrid spectrogram and waveform source separation. In *Proceedings of the ISMIR 2021 Workshop on Music Source Separation*, 2021.

[45] Simon Rouard, Francisco Massa, and Alexandre Défossez. Hybrid transformers for music source separation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[46] 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.

[47] Max Bain, Jaesung Huh, Tengda Han, and Andrew Zisserman. Whisperx: Time-accurate speech transcription of long-form audio. *arXiv preprint arXiv:2303.00747*, 2023.

[48] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

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

[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] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[52] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. Fréchet Audio Distance: A Reference-Free Metric for Evaluating Music Enhancement Algorithms. In *Proc. Interspeech 2019*, pages 2350–2354, 2019.

[53] 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.

[54] ByteDance team. Haimian music. *URL https://www.haimian.com*, 2025.

[55] Junmin Gong, Wenxiao Zhao, Sen Wang, Shengyuan Xu, and Jing Guo. Ace-step: A step towards music generation foundation model. `https://github.com/ace-step/ACE-Step`, 2025. GitHub repository.

[56] Luca Lanzendörfer, Florian Grötschla, Emil Funke, and Roger Wattenhofer. Disco-10m: A large-scale music dataset. *Advances in Neural Information Processing Systems*, 36, 2024.

[57] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Ismir*, volume 2, page 10, 2011.

[58] 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.

[59] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[60] Stable audio: Fast timing-conditioned latent audio diffusion. [Online]. `https://stability.ai/research/stable-audio-efficient-timing-latent-diffusion`.

[61] Haohe Liu, Xuenan Xu, Yi Yuan, Mengyue Wu, Wenwu Wang, and Mark D Plumbley. Semanticodec: An ultra low bitrate semantic audio codec for general sound. *IEEE Journal of Selected Topics in Signal Processing*, 2024.

[62] Shengpeng Ji, Ziyue Jiang, Wen Wang, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Xize Cheng, Zehan Wang, Ruiqi Li, et al. Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling. In *The Thirteenth International Conference on Learning Representations*, 2025.

[63] Zhen Ye, Peiwen Sun, Jiahe Lei, Hongzhan Lin, Xu Tan, Zheqi Dai, Qiuqiang Kong, Jianyi Chen, Jiahao Pan, Qifeng Liu, et al. Codec does matter: Exploring the semantic shortcoming of codec for audio language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25697–25705, 2025.

[64] Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap learning audio concepts from natural language supervision. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

A Project Contributors

- Project Sponsors: Dong Yu, Zhiyong Wu
- Core Contributors:
  - Data Collection & Washing: Wei Tan, Hangting Chen, Jianwei Yu
  - Music Encoder: Shun Lei, Hangting Chen, Haina Zhu
  - Diffusion & VAE: Yaoxun Xu, Zhiwei Lin, Hangting Chen
  - LLM Pre-train & Post-train: Shun Lei, Wei Tan, Huaicheng Zhang, Chenyu Yang, Hangting Chen, Jianwei Yu
  - Music Caption: Yixuan Zhang, Wei Tan
- Other Contributors: Chuan Lin, Rongzhi Gu*, Ruoning Hong, Shang Gao, Shulan Zhang, Yi Luo*, Yizhi Zhou* ($*$ contributed while working in Tencent AI Lab)

B Training and Implementation Details

B.1 LeLM

Our LeLM consists of a language model and an AR decoder. The language model is a 28-layer Transformer with 1536 hidden size. The AR decoder is a 12-layer Transformer with 1536 hidden size. We provide detailed hyperparameter settings about this model configuration in Table 4. We collected approximately 110,000 hours of songs from the internet for model training, comprising part of the DISCO-10M [56], Million Song Dataset [57], and some copyrighted in-house data.

During training, we used 32 NVIDIA H20 GPUs, with a batch size of 2 for each GPU to train the LeLM for 265K steps. Of these, 200k steps are required for the pre-training, with 60k steps for modular extension training, then 5k steps for multi-preference alignment. Adam optimizer is used with $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 10^{-9}$ and follow the same learning rate schedule in [58]. Consistently, top-k sampling is adopted for inference, in which k and temperature are set to 50 and 0.9, respectively.

Table 4: Hyper-parameters of LeLM.

Hyper-parameter		Value
Language model	Encoder Layers	28
	Hidden Size	1536
	Attention Head	12
	Feed-Forward Dim	8960
	Max Context Length (in #tokens)	8196
AR Decoder	Decoder Layers	12
	Hidden Size	1536
	Attention Head	12
	Feed-Forward Dim	8960
	Max Context Length (in #tokens)	10000
Total Number of Parameters		2136M

B.2 Music Codec

The Music Codec consists of four main components. First, the MuEncoder [40] extracts representations specifically designed for music. Next, the RVQ module discretizes these representations. The Diffusion Transformer (DiT) then transforms the discretized representations into Variational AutoEncoder (VAE) representations, which are finally decoded back into waveforms by the VAE. In the following parts, we will provide a detailed description of the architecture and training parameters of both the DiT and the VAE.

B.2.1 DiT

Building upon the GPT-2 [3] architecture, we developed the core framework of DiT [59], incorporating rotary positional encoding to enhance the model's capacity for sequence modeling. In addition, we introduced a timestep feature injection mechanism, which deeply integrates temporal information into the inference process, thereby aligning more effectively with the denoising dynamics of diffusion models.

To address the challenge of limited GPU memory during long audio generation, we adopted a chunk-wise cascaded inference strategy. Specifically, inspired by MuCodec's approach [40] to maintaining coherence, we use the tail features of the previous audio chunk as a prompt to guide the generation of the current chunk, ensuring smooth transitions between musical segments. Furthermore, we implemented a masking mechanism to distinguish between prompt regions and regions to be generated. By concatenating the prompt features, mask indicators, and the audio codec representations as the model's input, we significantly enhance the model's ability to generate long, coherent sequences.

Recognizing the distinct characteristics of mixed token and dual-track token audio representations, we designed differentiated network structures for each. In the mixed token mode, audio is first processed by a MuEncoder [40] and then discretized via RVQ to obtain a unified representation. In contrast, the dual-track mode involves separating vocals and accompaniment, each of which is independently encoded and discretized, resulting in two distinct representations. To accommodate these differing requirements, we tailored the model configurations accordingly: for the mixed token mode, we employed 1000 positional encodings, 39 transformer layers, 30 attention heads, and an embedding dimension of 1200; for the dual-track mode, we used 1000 positional encodings, 16 transformer layers, 20 attention heads, an embedding dimension of 2200, and set the inner layer dimension to 4400.

We trained our DiT model using a large-scale dataset of 300,000 music samples sourced from the internet, incorporating subsets of DISCO-10M [56], the Million Song Dataset [57], and our own in-house collections. From this dataset, we randomly selected 10,000 music samples to serve as our test set. For each training epoch, we randomly sampled 100,000 audio clips from the full training set, while 200 samples were drawn from the test set for evaluation during that epoch. For optimization, we employed the AdamW optimizer, setting the learning rate to 3×10^{-5} and the weight decay to 1e-8. The optimizer's beta parameters were configured as β_1 = 0.9 and β_2 = 0.999, with an epsilon value of 10^{-8} to ensure numerical stability. Training was conducted with a batch size of 3 and gradient accumulation over 3 steps to effectively manage memory usage. For both the Music Codec (Mixed) and Music Codec (Dual-Track) models, we utilized 8 40GB NVIDIA A100 GPUs for each configuration, training each model's DiT for 300,000 steps.

B.2.2 VAE

We retrained a 48kHz stereo Wave VAE based on Stable Audio [60], with a latent representation dimension of 64. Specifically, we set the strides of the encoder and decoder to 2, 4, 4, 6, and 10, resulting in a downsampling rate of 1920. Consequently, the original 48kHz audio is downsampled to a final frame rate of 25Hz, which is consistent with the frame rate of the muencoder. Apart from these adjustments, all other configurations—including model parameters for both the generator and discriminator, as well as training hyperparameters—remain identical to those used in Stable Audio[5].

The Wave VAE was trained for 1,000k steps on the same music dataset as LeLM on 8 40GB NVIDIA A100 GPUs. During DiT training, the parameters of the Wave VAE were kept frozen, and its latent representations of music were used as the training targets for DiT.

C Results of the Music Reconstruction

C.1 Reconstruction quality

To comprehensively and thoroughly evaluate the performance of Music Codec, we selected several representative and high-performing audio codecs for comparison, including SemantiCodec [61], WavTokenizer [62], XCodec [63], and MuCodec [40]. Following the experimental setup of MuCodec, we focused on two key aspects: vocals (measured by speaker embedding similarity, SPK_SIM, and word error rate, WER) and accompaniment (measured by the VISQOL metric). This allows

[5] `https://github.com/Stability-AI/stable-audio-tools/blob/main/stable_audio_tools/configs/model_configs/autoencoders/stable_audio_2_0_vae.json`

for a more detailed assessment of Music Codec's performance across different dimensions. The corresponding experimental results are presented in Table 5.

Table 5: Comprehensive comparison of Codec reconstruction quality across bitrates.

Method	CodeBook	Tokenrate (tps)	Bitrate (kbps)	VISQOL ↑	SPK_SIM ↑	WER (%)
Original music	—	—	—	—	—	10.92
SemantiCodec	1 x 32768	25	0.375	1.92/1.92	0.52	120.17
	1 x 16384	100	1.40	1.96/1.96	0.68	55.17
WavTokenizer	1 x 4096	40	0.48	2.93/2.93	0.49	101.49
	1 x 4096	75	0.90	3.05/3.05	0.56	86.19
XCodec	1 x 1024	50	0.50	3.04/3.04	0.53	85.10
	2 x 1024	50	1.00	3.30/3.30	0.79	55.37
	4 x 1024	50	2.00	3.38/3.38	0.63	36.32
	8 x 1024	50	4.00	3.58/3.58	0.72	26.42
MuCodec	1 x 16384	25	0.35	3.17/3.18	0.75	36.21
	4 x 10000	25	1.33	3.45/3.46	0.87	24.26
Music Codec (Mixed)	1 x 16384	25	0.35	3.27/3.27	0.78	38.22
	2 x 16384	25	0.70	3.34/3.34	0.82	33.43
	4 x 16384	25	1.40	3.52/3.53	0.84	28.92
Music Codec (Dual-Track)	16384+16384	25	0.70	3.43/3.44	0.82	31.54

In low-bitrate scenarios (bitrate < 0.5 kbps), both MuCodec and Music Codec demonstrate clear advantages over WavTokenizer, SemantiCodec, and XCodec. Whether in terms of preserving vocal characteristics or reconstructing background quality, MuCodec and Music Codec consistently achieve superior scores. For example, at a bitrate of 0.35 kbps, both Music Codec (Mixed) and MuCodec significantly outperform other methods in terms of VISQOL, SPK_SIM, and WER, fully showcasing their reconstruction capabilities at extremely low bitrates.

As the bitrate increases, the reconstruction performance of XCodec, WavTokenizer, and SemantiCodec improves in both vocal and background dimensions. However, they still lag behind MuCodec and Music Codec at comparable bitrates. For instance, at around 0.9 kbps, WavTokenizer shows improvements in VISQOL and SPK_SIM, but its WER remains higher than that of MuCodec and Music Codec. Similarly, XCodec achieves a VISQOL of 3.58 at 4.0 kbps, yet its WER is still higher than that of Music Codec at much lower bitrates. These results indicate that MuCodec and Music Codec not only enhance audio quality and intelligibility but also maintain better bitrate efficiency.

A closer comparison between MuCodec and Music Codec (Mixed) reveals that both employ a hybrid representation to model vocals and background, and their overall performance is similar at the same bitrate (e.g., 0.35 kbps). However, Music Codec (Mixed) is slightly inferior to MuCodec in terms of the VISQOL metric. This can be attributed to MuCodec's use of the Mel-spectrogram as an intermediate representation. Compared to directly mapping latent features to waveforms, using the Mel-spectrogram as a bridge provides stronger expressive power for audio quality reconstruction.

Furthermore, it is noteworthy that Music Codec (Dual-Track) achieves even better performance than Music Codec (Mixed) at the same bitrate of 0.70 kbps across all metrics. Although both operate at the same bitrate, the dual-Track approach explicitly models vocals and background separately, which allows for more accurate reconstruction of audio details and structure compared to the hybrid approach. This is also one of the main reasons why we ultimately adopt dual-track tokens as the prediction target in the LeVo system.

C.2 Reconstruction efficiency

In this experiment, we conducted a comparative analysis of the inference speed between MuCodec and Music Codec (Dual-Track), with detailed results presented in Table 6. We adopted the Real-

Time Factor (RTF) as the evaluation metric and divided the audio reconstruction process into three stages: first, Wav2Code, where the Codec Encoder transforms audio into discrete codes; second, Code2Latent, in which the DiT model performs multi-step denoising to convert the codes into latent representations; and finally, Latent2Wav, where the VAE decoder reconstructs the audio from the latent representations. To ensure a fair comparison, we set the number of inference steps for both models to 50.

Table 6: Comparison of Inference Efficiency between MuCodec and Music Codec (Dual-Track).

Method	DiT Param	Real-Time Factor			
		Wav2Code	Code2Latent	Latent2Wav	Total
MuCodec	743M	0.0201	0.5576	0.1001	0.6778
Music Codec (Dual-Track)	770M	0.0183	0.0680	0.0039	0.0902

As shown in Table 6, MuCodec and Music Codec (Dual-Track) have very similar DiT parameter sizes, at 743M and 770M, respectively. In the Wav2Code stage, their RTFs are 0.0201 and 0.0183, indicating almost identical efficiency in the encoder component. However, in the Code2Latent stage, MuCodec's RTF reaches 0.5576, while Music Codec (Dual-Track) achieves a much lower 0.0680—an almost eightfold difference—demonstrating a significant advantage for Music Codec (Dual-Track) in generating latent representations through denoising. Furthermore, in the Latent2Wav stage, MuCodec's RTF is 0.1001, compared to just 0.0039 for Music Codec (Dual-Track), a similarly striking gap. This improvement can be largely attributed to the fact that Music Codec (Dual-Track) establishes a direct mapping from latent representations to audio, bypassing the intermediate mel-spectrogram representation and thus greatly reducing inference time.

Overall, Music Codec (Dual-Track) significantly outperforms MuCodec in terms of total inference speed, with a total RTF of only 0.0902 compared to MuCodec's 0.6778—a nearly sevenfold difference. This demonstrates that, given similar parameter sizes, Music Codec (Dual-Track) offers a clear advantage in inference efficiency. Moreover, as shown in the results of Table 1, its reconstruction performance is highly comparable to that of MuCodec. This combination of speed and quality is the primary reason we have chosen Music Codec (Dual-Track) as the core decoding module in the LeVo system.

D Detailed Experimental Settings

For evaluation, we generate 20 distinct lyrics and accompanying text descriptions with a large language model, and select 20 unseen music clips as audio prompts. For every lyric we produced up to three song variants, depending on system capabilities: (i) lyric-only, (ii) lyric + text description, and (iii) lyric + audio prompt. All metrics were computed over the entire set of generated songs.

D.1 Details in objective evaluations

Here, we provide details of the objective evaluations.

FAD Fréchet Audio Distance (FAD) is used to evaluate the generation fidelity of music. We calculate FAD based on the distribution distance between the feature of the generated audio and MusicCaps [14] test set using audioldm_eval[6].

PER Phoneme Error Rate (PER) is used to evaluate the lyrics alignment and intelligibility of songs. To calculate PER, the vocal track is first extracted by Demucs, and then Whisperlarge-v2 [47] is utilized for lyric recognition. Using the clean vocal track rather than the song audio improves recognition accuracy. Whisperlarge-v2 is one of the state-of-the-art automatic speech recognition (ASR) models. It not only achieves high performance for speech but is also robust in recognizing singing voice. Considering singing often involves homophones or elongated vowels that do not alter

[6]https://github.com/haoheliu/audioldm_eval

phonetic content. Word-level (WER) or character-level (CER) metrics, therefore, penalize many errors that are inaudible to listeners. By operating at the phoneme level, PER captures intelligibility more faithfully and correlates better with human ratings of lyric clarity.

MuQ-T and MuQ-A To quantify how faithfully LeVo follows user prompts, we report two embedding-based similarity scores derived from MuQ-MuLan [41], a contrastive music–language pre-training model.

- **MuQ-T**: similarity between the song and its text description.
- **MuQ-A**: similarity between the song and its audio prompt.

It is important to note that while the CLAP [64] is commonly used for similarity calculations, the training set of CLAP is dominated by non-vocal audio events, leading to inaccurate similarity comparisons. For a clean evaluation, MuQ-T is calculated on songs generated only from the text description, and MuQ-A on songs generated only from the audio prompt, avoiding conflicts between mismatched prompt modalities.

Meta Audiobox-Aesthetic Measures the perceived musical aesthetics by leveraging advanced neural networks, including content enjoyment (CE), content usefulness (CU), production complexity (PC), and production quality (PQ).

D.2 Details in subjective evaluations

Table 7: Subjective results of comparison and ablation systems for song generation on OVL, MEL, and HAM with 95% Confidence Intervals. The overall first and second results are marked with **bold** and <u>underline</u>, respectively.

Models	**MOS ↑**		
	OVL	**MEL**	**HAM**
Suno-V4.5	$\mathbf{3.59 \pm 0.054}$	$\mathbf{4.10 \pm 0.042}$	$\mathbf{3.93 \pm 0.033}$
Haimian	3.05 ± 0.044	3.51 ± 0.046	3.55 ± 0.049
Mureka-O1	$\underline{3.42 \pm 0.047}$	3.88 ± 0.049	3.89 ± 0.030
YuE	2.45 ± 0.053	3.04 ± 0.052	2.94 ± 0.058
DiffRhythm	2.60 ± 0.054	3.18 ± 0.051	3.22 ± 0.066
ACE-Step	2.26 ± 0.065	3.02 ± 0.070	3.30 ± 0.073
SongGen*	2.91 ± 0.049	3.43 ± 0.053	3.44 ± 0.055
LeVo	$\underline{3.42 \pm 0.051}$	$\underline{3.93 \pm 0.044}$	$\underline{3.90 \pm 0.032}$
w/o Train stage 2	3.29 ± 0.044	3.76 ± 0.052	3.77 ± 0.044
w/o AR decoder	2.93 ± 0.053	3.44 ± 0.057	3.34 ± 0.073
w/o Dual-Track	3.25 ± 0.049	3.82 ± 0.048	3.84 ± 0.040
w/o DPO	3.18 ± 0.050	3.71 ± 0.061	3.76 ± 0.039

For the subjective evaluations, we focus on the performance of generated songs in the following six dimensions:

- **Overall Quality (OVL).** The overall musicality and naturalness of the generated song.
- **Vocal Melodic Attractiveness (MEL).** The beauty, smoothness, and appeal of the vocal melody.
- **Vocal-Instrument Harmony (HAM).** The coherence and integration between vocals and accompaniment.
- **Song Structure Clarity (SSC).** The clarity and distinction of song sections (e.g., verse, chorus, bridge)
- **Audio Sound Quality (AQ).** The clarity, absence of noise, dynamic range, and technical quality of the audio.

Table 8: Subjective results of comparison and ablation systems for song generation on SSC, AQ, and LYC with 95% Confidence Intervals. The overall first and second results are marked with **bold** and underline, respectively.

Models	**MOS ↑**		
	SSC	**AQ**	**LYC**
Suno-V4.5	$\mathbf{4.19 \pm 0.077}$	$\mathbf{4.00 \pm 0.068}$	3.17 ± 0.067
Haimian	3.62 ± 0.060	3.87 ± 0.040	$\underline{3.32 \pm 0.062}$
Mureka-O1	$\underline{4.14 \pm 0.077}$	3.87 ± 0.068	$\underline{3.32 \pm 0.056}$
YuE	3.53 ± 0.062	3.08 ± 0.055	2.41 ± 0.060
DiffRhythm	3.55 ± 0.060	3.09 ± 0.054	2.69 ± 0.050
ACE-Step	3.21 ± 0.092	2.36 ± 0.064	2.22 ± 0.060
SongGen*	3.66 ± 0.061	3.69 ± 0.054	2.84 ± 0.063
LeVo	4.09 ± 0.060	$\underline{3.96 \pm 0.038}$	$\mathbf{3.38 \pm 0.071}$
w/o Train stage 2	3.80 ± 0.052	$\underline{3.96 \pm 0.037}$	2.91 ± 0.072
w/o AR decoder	3.59 ± 0.044	3.71 ± 0.051	2.74 ± 0.058
w/o Dual-Track	3.96 ± 0.058	3.86 ± 0.038	3.18 ± 0.073
w/o DPO	3.97 ± 0.068	3.93 ± 0.041	3.18 ± 0.078

- **Lyrics Following Accuracy (LYC).** The accuracy and clarity with which the vocals follow the intended lyrics.

We conducted MOS (Mean Opinion Score) tests for all aspects, providing subjects with detailed descriptions, and reported both mean and CI95 scores of our MOS tests. The subjects present and rate the samples, and each subject is asked to evaluate on a scale of 1-5. Our MOS tests were conducted by 10 music professionals. All the screenshots of instructions for subjects have been shown in Figure 4-9. We paid at least the minimum wage in the country of these subjects. We tell the subjects that the data will be used in scientific research.

E Detailed Results with 95 % Confidence Intervals

To support the main claims of the paper, we report in Tables 7 and 8 the complete MOS scores together with their 95 % confidence intervals (CI95). Table 7 presents MOS for Overall quality (OVL), Vocal Melodic Attractiveness (MEL), and Vocal-Instrument Harmony (HAM). Table 8 lists MOS for Song Structure Clarity (SSC), Audio Sound Quality (AQ), and Lyrics Alignment Accuracy (LYC). For objective evaluations, the error bars are very small due to the large number of test samples. Specifically, at a 95% confidence level, the error margin for MuQ-T and MuQ-A scores across all models is less than 0.01, and for content enjoyment (CE), content usefulness (CU), and production quality (PQ), the error margin is less than 0.015. For production complexity (PC), the error margin is less than 0.05. The narrow intervals confirm that inter-rater variability is low and that the trends reported in Section 4 are statistically reliable:

- **Superiority of LeVo.** LeVo's MOS scores remain significantly higher than every open-source system across all six dimensions. The non-overlapping CIs corroborate LeVo's superiority.
- **Competitiveness with industry systems.** While Suno-V4.5 leads on most dimensions, its CIs overlap with (or are exceeded by) LeVo's on many dimensions, underscoring our claim of state-of-the-art alignment performance even against proprietary systems.
- **Effectiveness of each module.** Ablation rows show consistent, statistically meaningful drops. These non-overlapping intervals validate the contribution of every design choice.

F Contribution of Each Module to Audio Quality

To investigate how different components of LeVo contribute to the final audio quality, we conducted a Mean Opinion Score (MOS) listening test on 40 unseen songs. The evaluated audio included:

Table 9: Subjective results on audio quality generated by different modules within LeVo.

Models	AQ(Audio Quality)
GT	3.81
VAE Recon	3.79
Codec Recon	3.75
LeLM	0.21

(1) ground-truth recordings, (2) VAE-reconstructed audio, (3) Codec-reconstructed audio, and (4) LeVo-generated audio using the same lyrics. As shown in Table 9, the overall degradation in audio quality mainly arises from three sources: VAE reconstruction loss (0.02), Codec reconstruction loss (0.04), and LeLM modeling loss (0.09). Among these, the loss introduced by LeLM modeling is the most pronounced. While LeVo achieves higher perceived audio quality than models such as SongGen and LeVo w/o AR Decoder—both of which generate dual-track tokens—it still lags behind the Codec reconstruction, revealing room for improvement in the generative modeling stage. In addition, both the VAE and Codec modules contribute to minor degradation during the token-to-audio process.

G Smooth Transitions in Interpolation-based Multi-Preference Alignment Method

To further demonstrate the flexibility of our proposed interpolation-based multi-preference alignment method, we measured its behavior under four weight settings that progressively move from no preference control to a uniform blend of the three preferences. When the model is trained without DPO, it serves as the baseline for all subsequent comparisons. The results are presented in Table 10. Steering the model with a single preference—that is, assigning an interpolation coefficient of (1, 0, 0) for Lyric Alignment Preferences (Strategy 1), (0, 1, 0) for Prompt Consistency Preference (Strategy 2), or (0, 0, 1) for Musicality Preference (Strategy 3) enhances exactly the targeted dimension. Concretely, Strategy 1 reduces the phoneme error rate (PER) from 10.6% to 7.0%, Strategy 2 raises MuQ-T and MuQ-A to 0.34 and 0.83, respectively, and Strategy 3 delivers the highest Audiobox-Aesthetic scores. When two preferences are mixed with equal weights, the model exhibits smooth trade-offs: combining Strategy 1 and Strategy 2 slightly relaxes PER (from 6.5 % to 7.0 %) compared to using Strategy 1 alone, but yields noticeable gains in MuQ-T and MuQ-A; blending Strategy 1 with Strategy 3 preserves most of the PER benefit while further improving Audiobox-Aesthetic scores compared to using Strategy 1 alone; pairing Strategy 2 with Strategy 3 simultaneously maintains high MuQ scores and Audiobox-Aesthetic scores. Finally, the uniform three-way interpolation (0.33, 0.33, 0.33) produces consistent improvements across all metrics. Taken together, these results verify that by simply adjusting the inference-time interpolation coefficients (without retraining or additional fine-tuning), this approach can achieve a smooth transition in performance to satisfy diverse application needs more flexibly than conventional data-mixing approaches.

Table 10: Comparison of various preference combinations across multiple objective metrics. The overall first and second results are marked with **bold** and <u>underline</u>, respectively.

Models	**FAD ↓**	**MuQ-T ↑**	**MuQ-A ↑**	**PER ↓**	**Content Scores ↑**			
					CE	**CU**	**PC**	**PQs**
w/o DPO	**2.60**	0.31	0.82	10.6	7.70	7.86	5.89	8.39
Strategy 1	2.85	0.30	0.81	**6.5**	7.72	7.86	5.97	8.42
Strategy 2	2.89	**0.34**	**0.83**	10.3	7.75	7.87	5.96	8.43
Strategy 3	<u>2.63</u>	0.32	0.82	11.2	**7.78**	**7.93**	**6.16**	<u>8.45</u>
Strategy 1&2	2.91	0.33	**0.83**	<u>7.0</u>	7.67	7.83	5.93	8.40
Strategy 1&3	2.75	0.31	0.82	8.3	7.76	7.90	<u>6.10</u>	8.44
Strategy 2&3	2.73	**0.34**	**0.83**	10.9	7.75	<u>7.92</u>	6.01	8.43
Strategy 1&2&3	2.68	**0.34**	**0.83**	7.2	**7.78**	7.90	6.03	**8.46**

Table 11: Pearson Correlation between subjective evaluation results and Meta Audiobox-Aesthetic metrics.

	OVL	MEL	HAM	SSC	AQ	LYC
CE	0.32	0.44	0.28	0.07	0.31	-0.01
CU	0.26	0.36	0.16	0.02	0.27	-0.14
PC	-0.08	-0.14	0.32	0.44	-0.14	0.23
PQ	0.21	0.07	-0.06	0.01	0.15	-0.12

H Risks of Memorization and Content Leakage

We further assess the potential risks of memorization and content leakage in LeVo. To examine whether the model inadvertently reproduces training data, we randomly selected 40 songs from the training set and generated corresponding outputs using the same input conditions. We then evaluated the similarities between the generated and original songs by computing the 5-gram overlap and Levenshtein distance similarity of the extracted dual-track token sequences, supplemented by manual inspection of melodic patterns. The quantitative results show extremely low similarity scores—0.0001 for 5-gram and 0.0012 for Levenshtein distance—and no perceptually similar melodies were identified through human evaluation. These results suggest that LeVo does not memorize or directly replicate its training data, even when prompted with identical inputs.

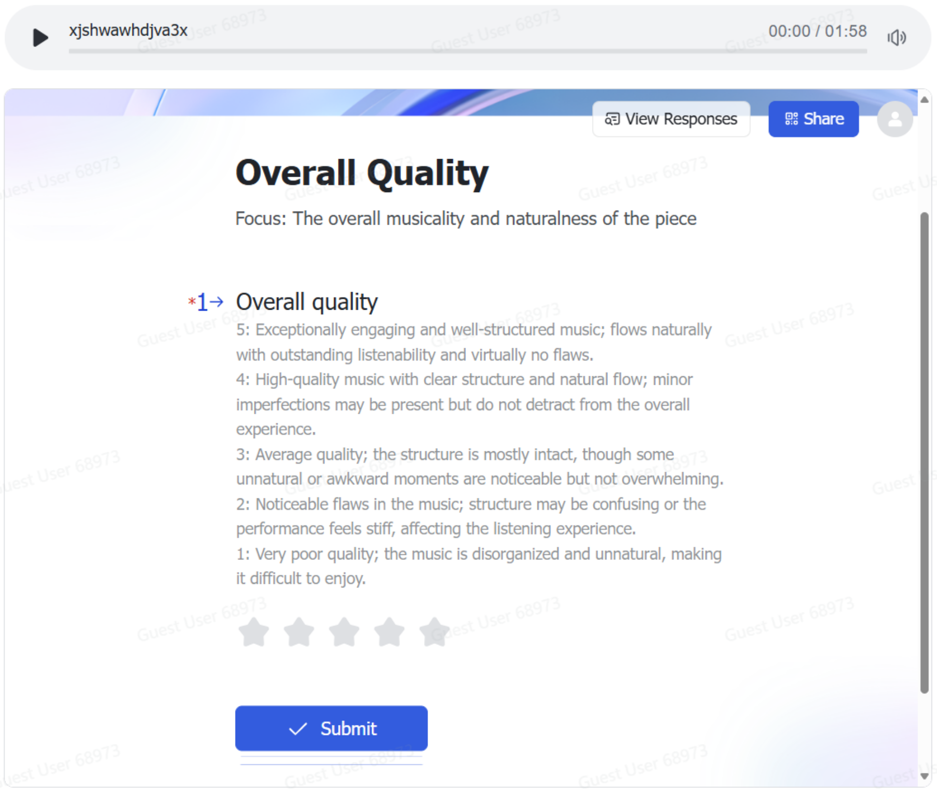
We attribute this low memorization tendency to two main factors. First, the relatively modest model scale (2B parameters) compared to the large training corpus of 2 million songs reduces the likelihood of overfitting to specific samples. Second, the multi-preference alignment stage following pre-training serves as an additional regularization process, further mitigating memorization by optimizing the model under diverse alignment objectives rather than maximizing likelihood on training data. Together, these results indicate that LeVo maintains a low risk of content leakage, supporting its reliability for safe and responsible music generation.

I Correlation between Human Evaluation Results and Objective Metrics

To better analyze the limitations of current objective metrics in music generation, we compared subjective and objective evaluation results. Overall, we observed a noticeable discrepancy between human judgments and automatic metrics. To quantify this relationship, we calculated the Pearson Correlation Coefficients between human evaluation results and objective scores.

A weak correlation was found between FAD and AQ ($r = -0.24$), suggesting that distribution-level metrics such as FAD may fail to capture subtle acoustic differences that influence human perception. In contrast, PER and LYC exhibit a strong correlation ($r = -0.77$), indicating that ASR-based metrics effectively reflect lyric-following accuracy. Nevertheless, these metrics are still affected by factors such as background accompaniment and vocal pitch, which can introduce recognition errors.

For the Meta Audiobox-Aesthetic metrics and subjective evaluation results, the correlations were shown in Table 11. The results show that content enjoyment (CE) exhibits the strongest correlations with OVL (0.32), MEL (0.44), and AQ (0.31), suggesting that these objective metrics are most aligned with human enjoyment of music. Content usefulness (CU) shows similar but slightly weaker patterns. Production complexity (PC) correlates most strongly with HAM (0.32) and SSC (0.44), indicating sensitivity to variations in vocal and accompaniment elements. In contrast, production quality (PQ) shows only weak correlations with all subjective dimensions, reflecting its limited ability to capture human-perceived quality.


xjshwawhdjva3x

00:00 / 01:58

View Responses

Share

Overall Quality

Focus: The overall musicality and naturalness of the piece

*1→ Overall quality

5: Exceptionally engaging and well-structured music; flows naturally with outstanding listenability and virtually no flaws.

4: High-quality music with clear structure and natural flow; minor imperfections may be present but do not detract from the overall experience.

3: Average quality; the structure is mostly intact, though some unnatural or awkward moments are noticeable but not overwhelming.

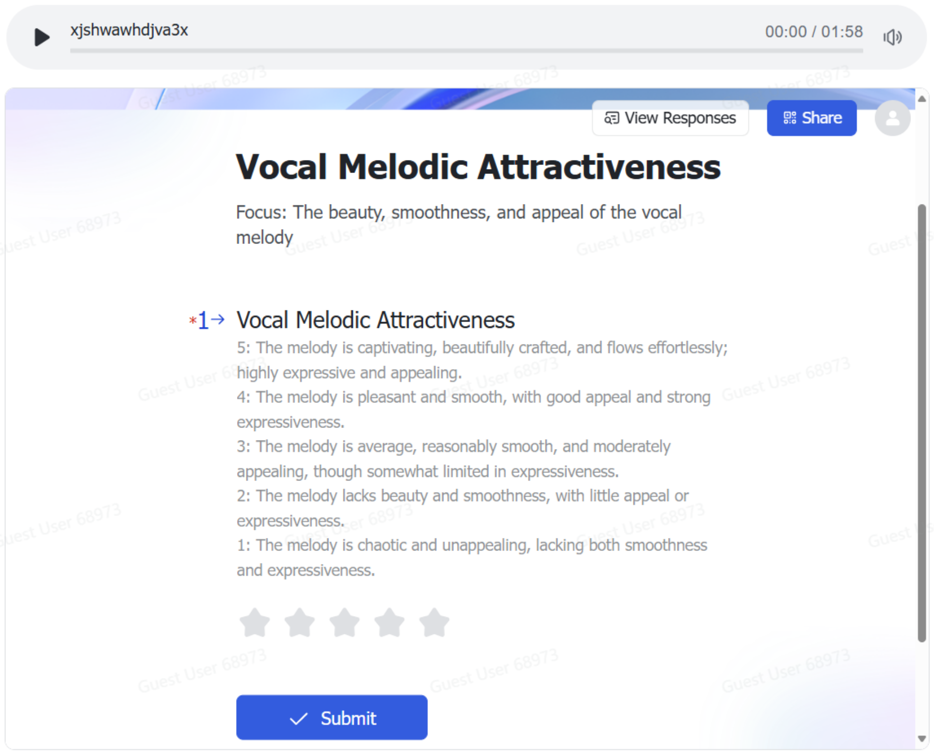
2: Noticeable flaws in the music; structure may be confusing or the performance feels stiff, affecting the listening experience.

1: Very poor quality; the music is disorganized and unnatural, making it difficult to enjoy.

☆☆☆☆☆

Submit

Figure 4: The screenshot of MOS test in overall quality.


xjshwawhdjva3x

00:00 / 01:58

View Responses

Share

Vocal Melodic Attractiveness

Focus: The beauty, smoothness, and appeal of the vocal melody

*1→ Vocal Melodic Attractiveness

5: The melody is captivating, beautifully crafted, and flows effortlessly; highly expressive and appealing.

4: The melody is pleasant and smooth, with good appeal and strong expressiveness.

3: The melody is average, reasonably smooth, and moderately appealing, though somewhat limited in expressiveness.

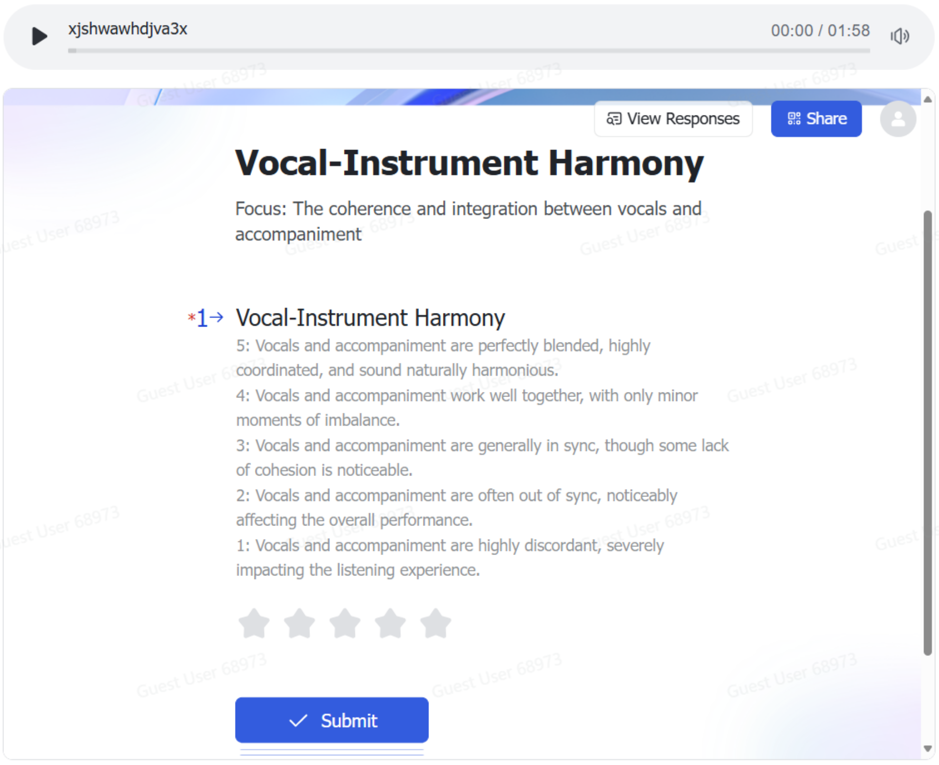
2: The melody lacks beauty and smoothness, with little appeal or expressiveness.

1: The melody is chaotic and unappealing, lacking both smoothness and expressiveness.

☆☆☆☆☆

Submit

Figure 5: The screenshot of MOS test in vocal melodic attractiveness.


xjshwawhdjva3x 00:00 / 01:58

View Responses Share

Vocal-Instrument Harmony

Focus: The coherence and integration between vocals and accompaniment

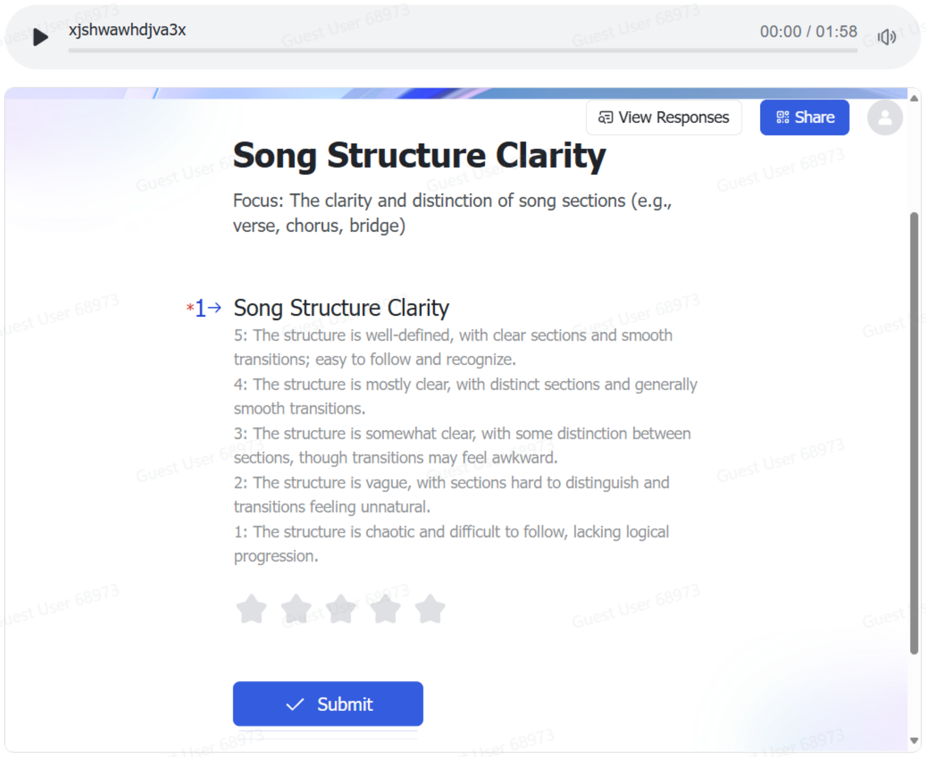
*1→ Vocal-Instrument Harmony

5: Vocals and accompaniment are perfectly blended, highly coordinated, and sound naturally harmonious.
4: Vocals and accompaniment work well together, with only minor moments of imbalance.
3: Vocals and accompaniment are generally in sync, though some lack of cohesion is noticeable.
2: Vocals and accompaniment are often out of sync, noticeably affecting the overall performance.
1: Vocals and accompaniment are highly discordant, severely impacting the listening experience.

☆ ☆ ☆ ☆ ☆

Submit

Figure 6: The screenshot of the MOS test in vocal-instrument harmony.


xjshwawhdjva3x 00:00 / 01:58

View Responses Share

Song Structure Clarity

Focus: The clarity and distinction of song sections (e.g., verse, chorus, bridge)

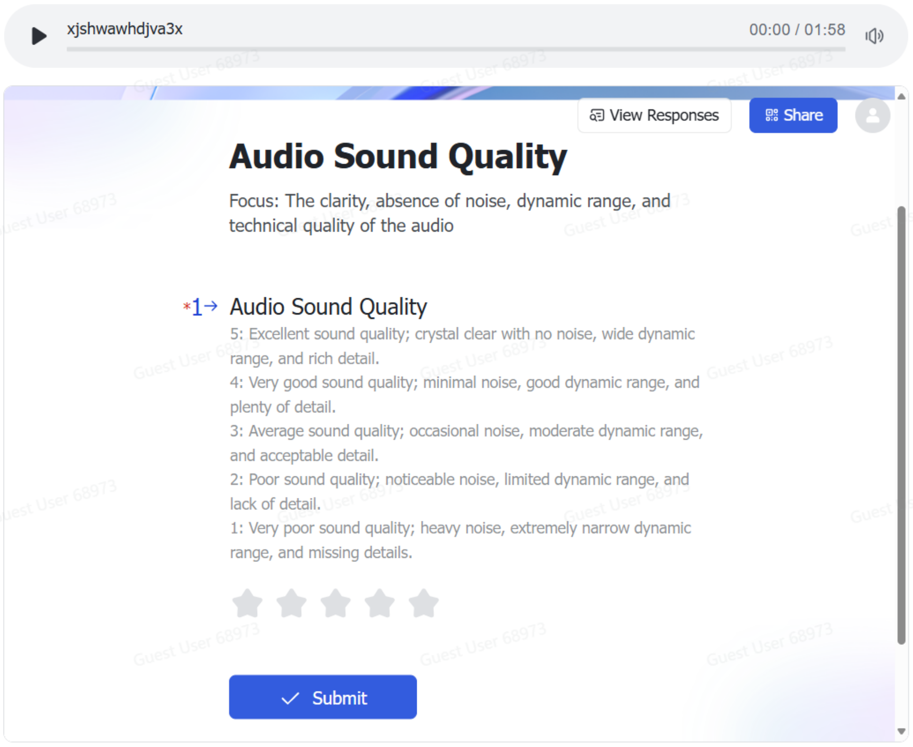
*1→ Song Structure Clarity

5: The structure is well-defined, with clear sections and smooth transitions; easy to follow and recognize.
4: The structure is mostly clear, with distinct sections and generally smooth transitions.
3: The structure is somewhat clear, with some distinction between sections, though transitions may feel awkward.
2: The structure is vague, with sections hard to distinguish and transitions feeling unnatural.
1: The structure is chaotic and difficult to follow, lacking logical progression.

☆ ☆ ☆ ☆ ☆

Submit

Figure 7: The screenshot of the MOS test in song structure clarity.



xjshwawhdjva3x 00:00 / 01:58

View Responses Share

Audio Sound Quality

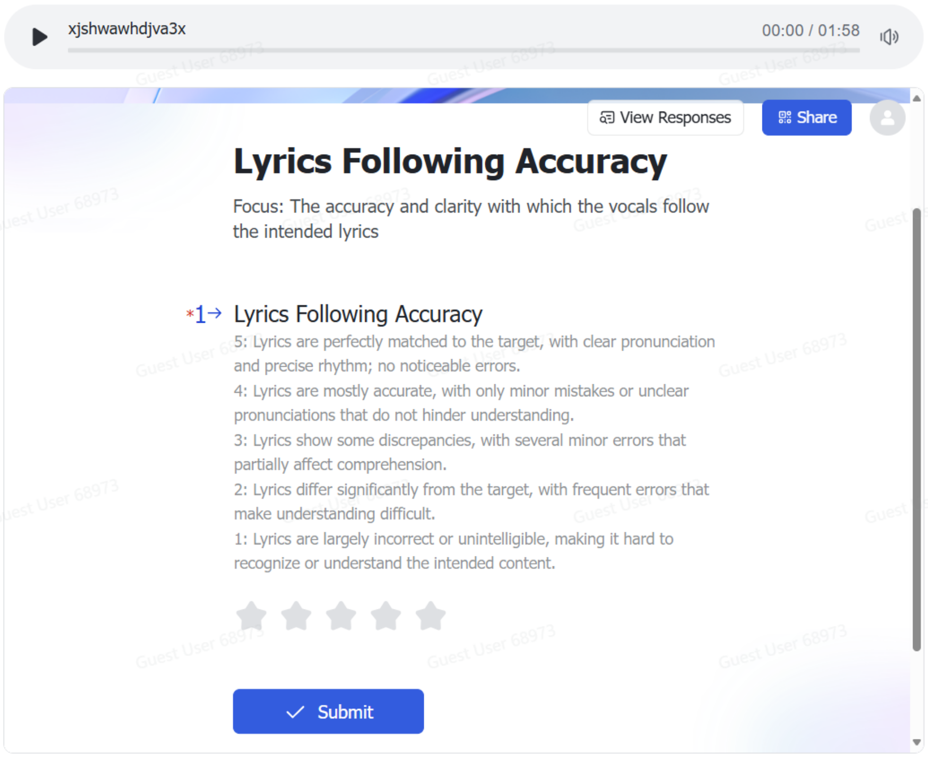
Focus: The clarity, absence of noise, dynamic range, and technical quality of the audio

*1→ Audio Sound Quality

5: Excellent sound quality; crystal clear with no noise, wide dynamic range, and rich detail.
4: Very good sound quality; minimal noise, good dynamic range, and plenty of detail.
3: Average sound quality; occasional noise, moderate dynamic range, and acceptable detail.
2: Poor sound quality; noticeable noise, limited dynamic range, and lack of detail.
1: Very poor sound quality; heavy noise, extremely narrow dynamic range, and missing details.

Submit

Figure 8: The screenshot of the MOS test in audio sound quality.



xjshwawhdjva3x 00:00 / 01:58

View Responses Share

Lyrics Following Accuracy

Focus: The accuracy and clarity with which the vocals follow the intended lyrics

*1→ Lyrics Following Accuracy

5: Lyrics are perfectly matched to the target, with clear pronunciation and precise rhythm; no noticeable errors.
4: Lyrics are mostly accurate, with only minor mistakes or unclear pronunciations that do not hinder understanding.
3: Lyrics show some discrepancies, with several minor errors that partially affect comprehension.
2: Lyrics differ significantly from the target, with frequent errors that make understanding difficult.
1: Lyrics are largely incorrect or unintelligible, making it hard to recognize or understand the intended content.

Submit

Figure 9: The screenshot of the MOS test in lyrics following accuracy.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We have ensured that the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We have thoroughly discussed the limitations of our work in Section 5.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

Justification: Our paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We have clearly and comprehensively described the architecture and training strategy of our model in Section 3, and provided detailed hyperparameters in Appendix B to facilitate replication of the model. We also open-source our code and checkpoint to make the results reproducible.

   Guidelines:

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

5. **Open access to data and code**

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

Answer: [Yes]

Justification: We have provided detailed instructions on data access and preparation in Section 4 and Appendix B. We have made the source code and checkpoints open-source, enabling anyone to reproduce all headline results.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We have provided all the training and test details in Section 4 and Appendix B.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We have reported 95% confidence intervals (CI95) for the experiments in Appendix E that support the main claims of the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Our experiments were conducted using GPUs, and detailed information about the computational resource is provided in Appendix B.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We confirm that our research conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We have discussed both potential positive societal impacts and negative societal imparts of the work performed in Section 5.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [Yes]

    Justification: We have described our safeguards for responsible release in Section 5, including but not limited to establishing proper constraints and usage guidelines

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: The creators or original owners of assets used in the paper have been properly credited, and the original papers have been cited or URLs have been provided.

    Guidelines:

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

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

13. **New assets**

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

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

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

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

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

    Answer: [Yes]

    Justification: We conducted crowdsourcing experiments, and in Appendix D, we have provided the full text of the instructions given to subjects along with screenshots. Details about compensation are also included.

    Guidelines:

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

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

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

    Answer: [Yes]

    Justification: We disclosed all potential risks to the subjects.

    Guidelines:

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

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

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

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