SongGen: A Single Stage Auto-regressive Transformer for Text-to-Song Generation

Zihan Liu [1 2] **Shuangrui Ding** [3] **Zhixiong Zhang** [4] **Xiaoyi Dong** [2 3] **Pan Zhang** [2] **Yuhang Zang** [2] **Yuhang Cao** [2]
Dahua Lin [3 2 5] **Jiaqi Wang** [2]

Abstract

Text-to-song generation, the task of creating vocals and accompaniment from textual inputs, poses significant challenges due to domain complexity and data scarcity. Existing approaches often employ multi-stage generation procedures, leading to cumbersome training and inference pipelines, as well as suboptimal overall generation quality due to error accumulation across stages. In this paper, we propose **SongGen**, a fully open-source, single-stage auto-regressive transformer designed for controllable song generation. The proposed model facilitates fine-grained control over diverse musical attributes, including lyrics and textual descriptions of instrumentation, genre, mood, and timbre, while also offering an optional three-second reference clip for voice cloning. Within a unified auto-regressive framework, SongGen supports two output modes: **mixed mode**, which generates a mixture of vocals and accompaniment directly, and **dual-track mode**, which synthesizes them separately for greater flexibility in downstream applications. We explore diverse token pattern strategies for each mode, leading to notable improvements and valuable insights. Furthermore, we design an automated data preprocessing pipeline with effective quality control. To foster community engagement and future research, we will release our model weights, training code, annotated data, and preprocessing pipeline. The code is available at `https://github.com/LiuZH-19/SongGen`.

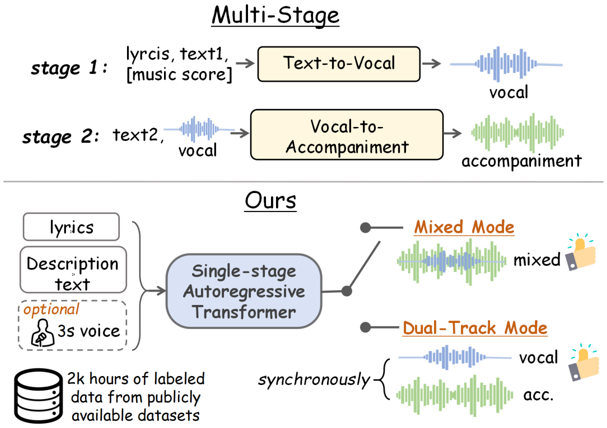

Figure 1. Multi-stage approaches generate vocals and accompaniment sequentially, leading to cumbersome pipelines and cumulative errors. SongGen simplifies this with a single-stage auto-regressive transformer that jointly models both tracks, supporting mixed and dual-track modes with better quality and efficiency.

1. Introduction

Songs, blending vocals with instrumental accompaniment, are a cornerstone of musical expression. Unlike purely instrumental music, songs uniquely capture human emotions through emotive lyrics and diverse melodies. However, creating a song is a complex, multi-stage process involving composition, instrumental arrangement, vocal performance, and more. This process requires substantial time and expertise, making it challenging for most individuals. With the rise of AI Generated Content (AIGC), creative fields have been revolutionized, extending from text and image generation (Rombach et al., 2022; Zhang et al., 2023; Achiam et al., 2023) to sophisticated artistic domains like music (Huang et al., 2019; Dhariwal et al., 2020; Ji et al., 2020). Building on these advancements, text-to-song generative models aim to transform natural language descriptions into full-song audio, making music creation more accessible and efficient.

Song generation presents greater complexity than speech or instrumental music generation (Lyth & King, 2024; Chen et al., 2024b; Liu et al., 2024; Copet et al., 2024). Unlike speech, singing spans a broader pitch range, incorporates a wider variety of expressive vocal techniques, and fol-

[1] Beihang University, Beijing, China [2] Shanghai AI Laboratory, Shanghai, China [3] The Chinese University of Hong Kong, Hong Kong, China [4] Harbin Institute of Technology, Harbin, China [5] CPII under InnoHK, Hong Kong, China. Correspondence to: Jiaqi Wang <wangjiaqi@pjlab.org.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

lows more dynamic rhythmic patterns. Moreover, song generation requires precise alignment between vocal and instrumental components to ensure musicality, harmonic consistency, and structural coherence—making it a uniquely challenging and underexplored problem. The scarcity of open-source data further limits research in this area.

Recent text-to-song approaches (Hong et al., 2024; Li et al., 2024a) decompose songs into separate vocal and accompaniment tracks and adopt multi-stage generation pipelines. As illustrated in Figure 1, these models first generate the vocal track from lyrics, then produce the accompaniment using natural language prompts alongside the generated vocals. However, such pipeline-based approach often fail to capture global optimality due to error accumulation across stages. This limitation is especially problematic for song generation. For instance, in genres like rap, vocal rhythm is tightly coupled with the instrumental beat. Generating vocals first without considering the underlying rhythm may result in rhythm misalignment. Conversely, in expressive genres such as ballads, where vocals typically guide the emotional flow, generating accompaniment first may constrain vocal expressiveness, resulting in rigid or disconnected performances. In both cases, pipeline approaches struggle to capture the intricate interplay between vocals and accompaniment. In addition, multi-stage generation results in cumbersome training and inference pipelines. Given these limitations in generation quality and efficiency, an important question arises: Is it possible for a single-stage model to achieve effective text-to-song generation?

In this paper, we introduce SongGen, a fully open-source, single-stage text-to-song generation model based on an auto-regressive transformer architecture. SongGen transforms lyrics and descriptive text into songs with harmonized vocals and accompaniment, allowing fine-grained control over instruments, genre, mood, timbre, and other musical elements. With a three-second reference vocal clip, it also supports zero-shot voice cloning. These user-defined controls are incorporated through modal-specific encoders, learnable projectors, and cross-attention mechanisms. SongGen offers two flexible generation modes: **mixed mode**, which blends vocals and accompaniment into a single output, and **dual-track mode**, which synthesizes them separately to facilitate professional post-production editing.

However, due to the sophisticated relationship between vocals and accompaniment in a song, jointly predicting them with natural expressiveness is a non-trivial task. To this end, we perform extensive explorations into output token patterns, yielding valuable insights. Specifically, (1) in **mixed mode**, while the model generates high-quality accompaniment, it struggles with natural-sounding vocals. Accompaniment, with higher energy and stable spectral distribution, tends to converge faster during training, whereas vocals, with higher semantic density and a lower signal-to-noise ratio due to overlap, present greater modeling challenges. This learning bias makes it difficult to generate vocals with clear lyrics, a problem typically addressed by decoupling and multi-stage methods. To mitigate this issue, we introduce an auxiliary vocal token prediction target, enhancing the model's focus on vocal features and significantly improving vocal clarity in mixed-token outputs. (2) In **dual-track mode**, vocals and accompaniment are treated as distinct yet interconnected sequences, generated in sync by a single transformer decoder. We explore various track combination patterns to maintain precise frame-level alignment. Experimental results indicate that the optimal pattern yields well-coordinated vocals and accompaniment, achieving quality on par with mixed-mode generation.

Moreover, the text-to-song generation community has long been constrained by data scarcity. To the best of our knowledge, no publicly available dataset currently includes paired audio, lyrics, and captions. To bridge this gap, we develop an automated pipeline for data cleaning, processing, and quality filtering, resulting in a high-quality dataset of 540K song clips spanning over 2,000 hours of audio.

To evaluate the effectiveness of the proposed SongGen framework, we conduct extensive experiments on the MusicCaps (Agostinelli et al., 2023) test set. The results demonstrate that SongGen outperforms the multi-stage baseline and generates songs with excellent musicality and vocal-instrument harmony, achieving performance that is competitive with the ground truth. Surprisingly, the generated songs feature expressive vocal techniques, such as vibrato, enhancing naturalness and authenticity. Our contributions can be summarized as follows:

- We introduce SongGen, a single-stage auto-regressive transformer for text-to-song generation, offering versatile control via lyrics, descriptive text, and an optional reference voice.
- SongGen supports both mixed and dual-track mode to accommodate diverse requirements. Our experiments provide valuable insights for optimizing both modes.
- By releasing the model weights, code, annotated data, and preprocessing pipeline, we aim to establish a simple yet effective baseline for future song generation research.

2. Related Work

2.1. Text-to-Music Generation

In recent years, significant progress has been made in text-to-music generation models, which use descriptive text as a condition for controllable music generation. Several works (Agostinelli et al., 2023; Copet et al., 2024) employ

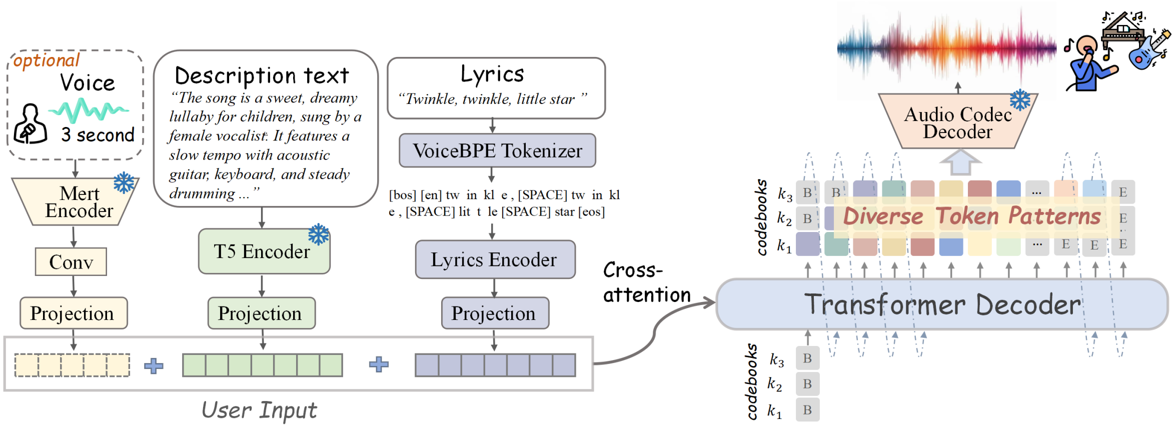
optional
Voice
3 second
Mert Encoder
Conv
Projection
Description text
"The song is a sweet, dreamy lullaby for children, sung by a female vocalist. It features a slow tempo with acoustic guitar, keyboard, and steady drumming ..."
T5 Encoder
Projection
Lyrics
"Twinkle, twinkle, little star "
VoiceBPE Tokenizer
[bos] [en] tw in kl e , [SPACE] tw in kl e , [SPACE] lit t le [SPACE] star [eos]
Lyrics Encoder
Projection
User Input
Cross-attention
Transformer Decoder
codebooks
k_3
k_2
k_1
B
E
...
Diverse Token Patterns
Audio Codec Decoder

Figure 2. Overview of SongGen: An auto-regressive transformer decoder generates audio tokens with diverse patterns, incorporating user-defined controls via cross-attention. The final song is synthesized from these tokens through the audio codec decoder.

transformer-based language models (LMs) (Vaswani et al., 2017) to model sequences of discrete tokens derived from audio codecs (Défossez et al., 2022; Zeghidour et al., 2021; Yang et al., 2023; Kumar et al., 2024). Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Kingma et al., 2021), another competitive class of generative models, have also attained impressive results in music generation(Forsgren & Martiros, 2022; Chen et al., 2024a; Evans et al., 2024; Schneider et al., 2023; Huang et al., 2023; Liu et al., 2024). While most models generate a mixture of stems, MusicGen-Stem (Rouard et al., 2025) proposes a multi-stem generative model with three separate tracks (bass, drums and other), suggesting that disentangling components can facilitate generation and source editing. However, although all the models discussed above excel at generating high-quality instrumental music, they face significant challenges in producing realistic vocals.

2.2. Song Generation

Recently, a few studies have begun exploring song generation, a task that involves vocal composition, instrumental arrangement, and harmonious generation. One of the pioneering efforts is Jukebox (Dhariwal et al., 2020), which employs a multi-scale VQ-VAE to compress audio into discrete codes and models them using a cascade of transformer models. However, Jukebox offers limited style control, relying solely on genre tags and artist names, and suffers from long inference times. Recently, models like Melodist (Hong et al., 2024) and MelodyLM (Li et al., 2024a) have adopted multi-stage approaches to address the challenges of text-to-song generation. Melodist integrates singing voice synthesis with vocal-to-accompaniment (V2A) techniques, while MelodyLM improves upon Melodist by overcoming its reliance on music scores through a three-stage process: text-to-MIDI, text-to-vocal, and V2A. However, both approaches result in cumbersome training and inference procedures, and their corpus is limited to Mandarin pop songs, lacking diversity. Another model, SongCreator (Lei et al., 2024), utilizes a dual-sequence language model to capture the relationship between vocals and accompaniment. However, it lacks text-based control and produces vocals with limited clarity. Freestyle (Ning et al., 2024) focuses on generating rapping vocals from lyrics and accompaniment inputs but is constrained to a single musical style, with rap typically featuring simpler melodies. The concurrent work Yue (Yuan et al., 2025) achieves impressive results by scaling up both data and model size, adopting a track-decoupled next-token prediction and a two-stage causal LM framework. In contrast, we explore more diverse and efficient token pattern designs within a single-stage transformer. Although industry tools like Suno[1] and Udio[2] have recently emerged for song generation, neither has disclosed their methodologies or expanded into broader controllable generation tasks. SeedMusic (Bai et al., 2024) leverages both auto-regressive language modeling and diffusion approaches to support song generation. However, SeedMusic is not open-source and relies on a large proprietary dataset, making a fair comparison with our fully open model unfeasible.

3. Methodology

3.1. Overview

The objective of this paper is to guide the generation of a song using a text description, lyrics, and an optional reference voice. As illustrated in Figure 2, SongGen is composed of an auto-regressive transformer decoder with an off-the-shelf neural audio codec. The transformer decoder predicts a sequence of audio tokens, allowing control through user inputs via cross-attention. The final song is synthesized from these tokens using the codec decoder. In the subsequent section, we will elaborate on the details of SongGen. Sec-

[1] https://suno.com/
[2] https://www.udio.com/

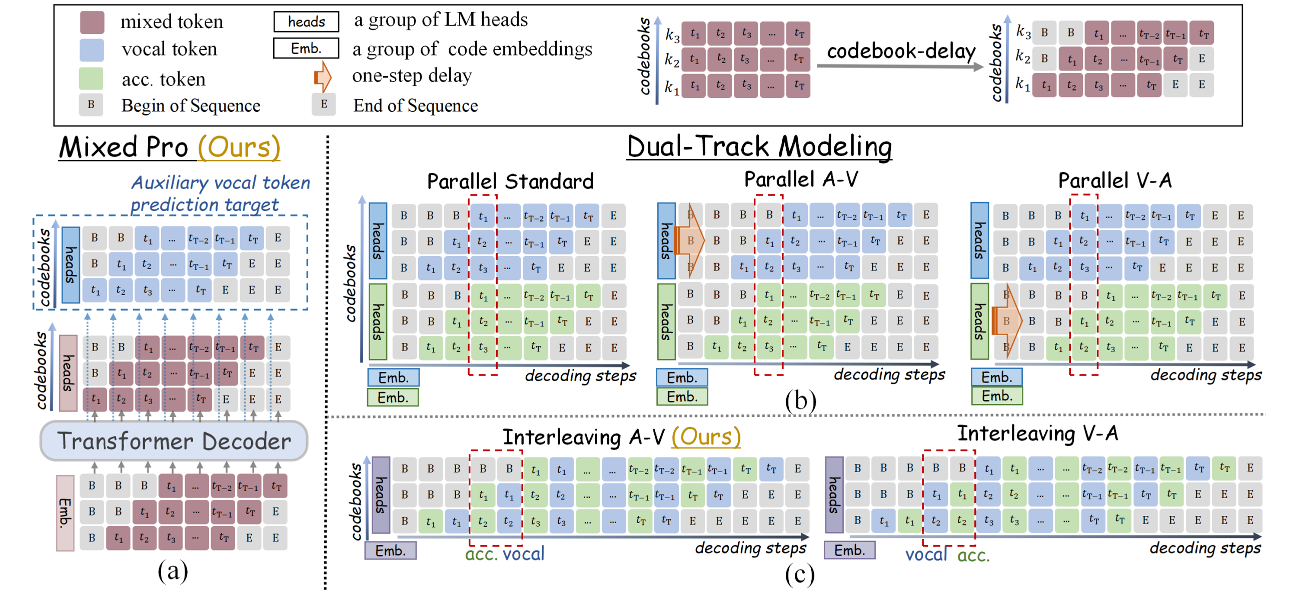

Figure 3. Illustration of token patterns for different generation modes. The codebook-delay pattern (from MusicGen) is applied to every audio token. (a) **Mixed Pro**: Directly decoding mixed tokens, with an auxiliary vocal token prediction target to enhance vocal learning. **Dual-track mode:** (b) Parallel: Vocal and accompaniment tokens are concatenated along the codebook dimension, with three track order variants. (c) Interleaving: Tokens from both tracks are interleaved along the temporal dimension, with two track order variants.

tion 3.2 will introduce the two generation modes supported by our unified framework: mixed mode and dual-track mode. Section 3.3 will discuss the lyric, voice, and text conditions. Section 3.4 will outline our data processing pipeline and quality filtering metrics. Section 3.5 will present our training scheme for progressively enhancing model performance.

3.2. Auto-regressive Codec Language Modeling

3.2.1. Audio Tokenization

The effectiveness of the audio tokenizer is critical to the success of transformer-based song generation. Our framework is compatible with mainstream Codec designs. In experiments, we employ X-Codec (Ye et al., 2024), an audio codec based on Residual Vector Quantizer (RVQ) (Zeghidour et al., 2021), to produce discrete audio tokens. It utilizes $N_q = 8$ codebooks, each with a codebook size of $K = 1024$. Given an audio signal $X \in \mathbb{R}^{d \cdot f_s}$, where d is the audio duration and $f_s = 16$ kHz is the sampling rate, X-Codec encodes and quantizes X into a sequence of token vectors $\mathbf{S} = [\mathbf{s}^1, \mathbf{s}^2, \dots, \mathbf{s}^T] \in \mathbb{R}^{N_q \times T}$, where $T = d \cdot f_r$ and $f_r = 50$ HZ is the frame rate. Each vector $\mathbf{s}^t = [s^{1,t}, s^{2,t}, \dots, s^{N_q,t}]$ consists of N_q codes, with $s^{k,t}$ taking integer values from 0 to $K-1$ for $k \in [1, N_q]$. We apply the codebook-delay pattern (Copet et al., 2024) to handle the multiple code sequences within a single transformer decoder architecture. Figure 3 at the top-right corner illustrates this process for the case of $N_q = 3$, where a one-step delay is maintained between adjacent sequences from different codebooks. After applying the delay pattern, the resulting code sequences are denoted as $\hat{S} \in \mathbb{R}^{N_q \times T'}$.

3.2.2. Mixed Mode

In mixed mode generation, we directly use the mixed audio tokens $\hat{\mathbf{S}}_{\text{mixed}}$, which are encoded by X-Codec from mixed audio (i.e. raw audio), as the output target. For each step, the vector of audio tokens from N_q codebooks are embedded using a group of N_q learnable embedding matrices, and then summed up to form the decoder input. Additionally, a sinusoidal positional embedding is added at each step. The last hidden state of decoder is passed to a group of N_q linear heads, with each head predicting the logits corresponding to its respective codebook.

During training, we employ the teacher-forcing scheme. Since each quantizer in the RVQ encodes the quantization error from the previous quantizer, earlier codebooks are more critical. Therefore, we compute a weighted sum of the losses from different codebooks, assigning higher importance to the losses from earlier codebooks:

$$\mathcal{L}_{\text{mixed}} = \sum_{k=1}^{N_q} w_k \cdot \mathcal{L}_{\text{mixed}}^k, \tag{1}$$

where k denotes the codebook index, and w_k represents the weight, satisfying $w_k \le w_j$ for $k < j$ and $\sum_{k=1}^{N_q} w_k = 1$. $\mathcal{L}_{\text{mixed}}^k$ is the cross-entropy loss for the k-th codebook.

However, this basic approach, referred to as "Mixed", presents challenges in producing coherent and clear vocals. In mixed audio, vocals suffer from a low signal-to-noise ratio because of overlap with the accompaniment. While the accompaniment typically exhibits higher energy and a more stable spectral distribution, the vocals tend to be sparser, more irregular, and prone to greater instantaneous frequency fluctuations. For example, vocals often feature rapid pitch changes to perform various singing techniques. Moreover, vocals carry more semantic meaning from the lyrics. When mixed audio is used as the training target, the model tends to prioritize the more predictable accompaniment, often neglecting the vocal features. Nevertheless, human perception is sensitive to the naturalness and clarity of vocals, making these aspects critically important in song generation.

Building on this, we propose a method called “Mixed Pro” that emphasizes vocal learning by introducing an auxiliary vocal token prediction target. As depicted in Figure 3 (a), we incorporate a dedicated group of linear heads to predict logits for vocal tokens. These tokens, encoded by X-Codec from the vocal track, are aligned frame-by-frame with the mixed tokens. The overall loss function is formulated as:

$$\mathcal{L}_{\text{mixed-pro}} = \mathcal{L}_{\text{mixed}} + \lambda \mathcal{L}_{\text{vocal}}, \tag{2}$$

where λ controls the contribution of the vocal loss to the total loss. It is important to note that these newly introduced vocal heads are used only during training to compute the auxiliary loss and do not affect inference.

3.2.3. Dual-track Mode

In dual-track generation mode, the two key components of a song—the vocal and the accompaniment—are separated, and SongGen synchronously generates both tracks within this unified framework. Considering the importance of harmony between vocals and accompaniment, we introduce two combination patterns, namely Parallel and Interleaving, to ensure frame-level alignment across the two tracks.

Parallel: Inspired by the stereo channel modeling of MusicGen (Copet et al., 2024), which simultaneously outputs audio tokens for two channels, we design a parallel pattern. As shown in Figure 3 (b), the accompaniment and vocal audio tokens are concatenated along the codebook dimension, with each step containing N_q vocal tokens and N_q accompaniment tokens. On the temporal dimension, we introduce three variants. In the “Standard” variant, the audio tokens for both tracks are strictly aligned frame by frame. The “Parallel (A-V)” variant delays the vocal tokens by one step relative to the accompaniment tokens. Thus, the vocal token prediction at each frame considers both the previous vocal token and the accompaniment token at the current frame. Conversely, in the “Parallel (V-A)” variant, the accompaniment tokens are delayed by one step relative to the vocal tokens. Two groups of code embeddings are used to separately embed the audio tokens for the two tracks. All embeddings are then averaged to form a combined input. Two groups of linear heads are employed to predict the audio tokens for each track. The training loss is defined as:

$$\mathcal{L}_{\text{parallel}} = \frac{1}{2}(\mathcal{L}_{\text{vocal}} + \mathcal{L}_{\text{acc}}), \tag{3}$$

where $\mathcal{L}_{\text{vocal}}$ and $\mathcal{L}_{\text{acc}}$ represent the individual losses for the vocal and accompaniment tracks, respectively. The calculation method is the same as in Equation 1.

Interleaving: In this pattern, the audio tokens of the two tracks are interleaved along the temporal dimension, as illustrated in Figure 3 (c). There are two variants: “Interleaving (A-V)”, where the accompaniment tokens precede the vocal tokens at each frame; and “Interleaving (V-A)”, where the vocal tokens precede the accompaniment tokens. In the ”Interleaving (A-V)” variant, each vocal token prediction at a given frame considers both the previous vocal token and the accompaniment token from the same frame, with the reverse for the ”Interleaving (V-A)” variant. In this pattern, only a single group of code embeddings and one group of heads are used. The training loss is calculated in the same way as in Equation 3.

Although the interleaving pattern requires longer sequence lengths than the parallel pattern, it provides a more effective approach to modeling the relationship between vocals and accompaniment. In the lower layers of the transformer, the interleaving pattern facilitates learning the interactions between the vocal and accompaniment tracks, while the higher layers focus on refining the distinct characteristics of each track. The attention visualizations in Figure 5 provide additional evidence for this. In contrast, the parallel pattern is unable to decouple the vocal and accompaniment information before reaching the heads.

3.3. Model Conditioning

Lyrics Conditioning. To address the challenge of data scarcity, we apply a 6681-token voice Byte-Pair Encoding (VoiceBPE) tokenizer (Casanova et al., 2024) to convert the lyrics C_{lyrics} into a sequence of phoneme-like tokens. Word-level tokenizers, like the T5(Raffel et al., 2020) tokenizer, lead to sparse training samples for each token embedding. In contrast, VoiceBPE not only enhances the model’s ability to generalize to unseen words but also adapts more effectively to the variations in phoneme duration and pitch range inherent in sung vocals. Subsequently, the lyrics embedding $E_{\text{lyrics}} \in \mathbb{R}^{T_l \times F_l}$ is obtained by passing the lyric tokens through a small transformer-based encoder (i.e., Lyrics Encoder) to extract critical pronunciation-related information. Here, T_l denotes the length of the lyric tokens, and F_l represents the dimensionality of the embedding.

Voice Conditioning. As demonstrated by the Marble (Yuan et al., 2023) benchmark, MERT (Li et al., 2024b), a music representation model, consistently achieves state-of-the-art performance in vocal technique detection and singer identification tasks. Consequently, we employ a frozen MERT encoder to generate robust voice feature embeddings, enabling control over vocal timbre and singing techniques. Specifically, we randomly select 3-second clips from vocal segments to serve as the voice condition input, denoted as C_{voice}. The outputs from MERT's 24 hidden layers and 1 output layer are aggregated via a 1D convolutional layer, yielding the voice embedding $E_{\text{voice}} \in \mathbb{R}^{T_v \times F_v}$, where T_v denotes the temporal length and F_v represents the feature dimensionality of the embedding.

Text Conditioning. Our text descriptions cover a wide range of musical attributes, including but not limited to the instruments used, musical emotion, tempo, genre, and the singer's gender, offering more depth than simple tags or short phrases. Given a description C_{text} matching the song, we apply a frozen FLAN-T5 (Chung et al., 2022) encoder to obtain the text embedding, denoted as $E_{\text{text}} \in \mathbb{R}^{T_t \times F_t}$.

The above three condition embeddings—E_{lyrics}, E_{voice}, and E_{text}—are each passed through their respective projection layers to obtain transformed embeddings, $\hat{E}_{\text{lyrics}}$, $\hat{E}_{\text{voice}}$, and $\hat{E}_{\text{text}}$. These embeddings are then concatenated along the temporal dimension:

$$E_{\text{cond}} = \hat{E}_{\text{voice}} \oplus \hat{E}_{\text{text}} \oplus \hat{E}_{\text{lyrics}} \in \mathbb{R}^{(T_v+T_l+T_t)\times D}, \quad (4)$$

where D denotes the dimension of the decoder hidden states. This concatenated embedding E_{cond} is used to control song generation via cross attention.

3.4. Automated Data Preprocessing Pipeline

To the best of our knowledge, there is currently no publicly available dataset for text-to-song generation that includes paired audio, lyrics, and captions. To address this gap, we develop an automated data annotation pipeline that incorporates several filtering strategies to ensure high-quality data. **(1) Data Source:** We collect 8,000 hours of audio from Million Song Dataset (MSD) (Bertin-Mahieux et al., 2011), Free Music Archive (FMA) (Defferrard et al., 2018) and MTG-Jamendo Dataset (Bogdanov et al., 2019). **(2) Source Separation:** We utilize Demucs (Rouard et al., 2023) to separate vocals and accompaniment from the original audio. **(3) Segmentation:** We employ a voice activity detection (VAD) tool (Gao et al., 2023) to detect voiced segments in the separate vocal tracks. Vocal, accompaniment, and mixed tracks are then sliced according to the VAD results, with an average clip duration of 15 seconds. Additionally, the energy of each clip is calculated as the sum of the squared amplitude over time, providing a measure of loudness. Clips with low energy in either the accompaniment or vocals are discarded. **(4) Lyric Recognition:** Lyric recognition accuracy is crucial for song generation, but it is challenging. Existing Automatic Speech Recognition (ASR) models, trained on speech data, struggle with the complexity and variability of sung vocals. Errors arise from two main factors: ASR limitations (misrecognitions and hallucinations); and inherently unclear vocal data, such as noise or genre-specific characteristics like those in rock music. To tackle this issue, we apply two ASR models, Whisper-large-v2 and Whisper-larger-v3 (Radford et al., 2022), to automatically transcribe the vocals and generate two lyric transcriptions. We compute the edit distance between them to assess quality, excluding clips with an edit distance greater than 20%, and retaining only those with relatively clearer vocals and higher recognition confidence. **(5) Captioning:** We use LP-MusicCaps-MSD (Doh et al., 2023) for MSD captions. For song clips without captions, we generate pseudo-captions using a music captioning model (Doh et al., 2023). The accuracy of the captions is evaluated by CLAP Score, which measures the alignment between audio and text with the official CLAP(Wu* et al., 2023) model. Samples with low CLAP scores are discarded, and any available original tags are added as a supplement. After preprocessing, the training dataset contains about 540K English-voiced clips, totaling around 2K hours of audio.

3.5. Training Scheme

Mixed Mode Training. Our mixed mode training consists of three key steps, aimed at progressively boost model performance. *Step 1: Modality Alignment.* We train the entire model using total paired data to align the modalities between the various conditioning inputs and the audio output. *Step 2 : Voice-Free Support.* To enable the model to function without a reference voice, we apply a 50% random drop to the reference voice input. To maintain the model's original capabilities, we freeze all modules related to user inputs and fine-tune only the transformer decoder. Once the decoder adapts, we unfreeze the entire model and fine-tune all parameters to optimize performance. *Step 3: High-Quality Fine-tuning.* The final stage refines the model using a carefully selected subset of data filtered by these quality metrics: edit_distance $\leq$ 5%, CLAP$_{src}$ $\geq$ 25%, energy $>$ 1000. This yields 100K high-quality pairs for fine-tuning, enabling the model to enhance the quality of audio by learning from cleaner, more relevant data.

Dual-track Mode Training. Our experiments revealed that training the dual-track mode from scratch is challenging. To address this, we initialize the dual-track model with the pre-trained mixed mode model after Step 1. *Step 1.5: Dual-Track Mode Adaptation.* After initialization, we freeze user input modules and fine-tune only the transformer decoder to adapt it to the new token pattern. Once the adaptation is complete, we unfreeze all model weights and proceed to fine-tune the entire model. The subsequent training steps mirror those of Steps 2 and 3 in the mixed mode.

Curriculum Learning for Codebook Loss Weight Adjustment. We propose a curriculum learning strategy to adjust the weights of codebook losses during training. Initially, the first three codebooks have weights of 0.25, while the rest are set to 0.05. This encourages the model to focus on the most important components first. As training progresses, the weights are gradually balanced, enabling the model to capture finer audio details step by step.

4. Experiments

4.1. Experimental setup

Baselines. To the best of our knowledge, no open-source text-to-song model is currently available. We therefore conduct a controlled comparison between single-stage and multi-stage approaches. The multi-stage baseline consists of two transformer models with the same architecture and training data as SongGen. Specifically, in Stage 1, the first model generates the vocal track from the lyrics, description, and a 3-second reference voice. In Stage 2, the second model generates the accompaniment, conditioned on the same textual inputs and the generated vocal (prepended into the decoder). The final song is then mixed from the two tracks. Additionally, we fine-tune Parler-tts (Lyth & King, 2024), a text-to-speech model that generates speech from both transcript and description texts, using our own training data. We also compare our model with Suno, a commercial product, using human evaluations.

Evaluation dataset and metrics. For the evaluation dataset, we filter the English-voiced song samples from MusicCaps benchmark (Agostinelli et al., 2023), yielding a test set of 326 samples, with the lyrics annotated by our preprocessing pipeline.

For automatic evaluations, Frechet Audio Distance (FAD) measures the generation fidelity; Kullback-Leibler Divergence (KL) evaluates conceptual similarity with the target audio; CLAP Score and CLaMP3 score (Wu et al., 2025) measures the alignment between the audio and the text description; Speaker Embedding Cosine Similarity (SECS) assesses the similarity of speaker identity; Phoneme Error Rate (PER) gauges adherence to the provided lyrics. Note that due to limitations in the ASR model, the PER values are higher than the actual error rate, but the relative differences remain meaningful. We also introduce content-based aesthetics metrics (Tjandra et al., 2025), covering Content Enjoyment (CE), Content Usefulness (CU), Production Complexity (PC), and Production Quality (PQ). For each method, we generate the audio five times with different random seeds and report the average metric.

For human evaluations, we employ Mean Opinion Score (MOS) tests, assessing five key aspects: overall quality (OVL.), focusing on musicality and naturalness; relevance to the text description (REL.); vocal quality, with an emphasis on clarity and intelligibility of the singing voice (VQ.); harmony between vocals and accompaniment (HAM.); and similarity to the original singer (SS.). The appendix B shows details of the evaluations.

4.2. Results of Text-to-song Generation

Tables 1 and 2 report the automatic and human evaluation results, respectively, for our mixed and dual-track models alongside several baselines. In both tables, the first 3-second vocal clip from the ground-truth is used as the reference voice for all our models and the multi-stage baseline.

Comparison with baselines. Although Parler-TTS excels in controllable text-to-speech, fine-tuning it for text-to-song is ineffective, as shown in the tables. This highlights the greater complexity of generating expressive vocals and musically coherent accompaniment in text-to-song tasks, compared to conventional speech synthesis.

Compared to the multi-stage baseline, our single-stage model outperforms it across both automatic and human evaluations. The only exceptions are the PER and SECS metrics, where the multi-stage model performs slightly better—due to its first-stage training being focused exclusively on clean vocal targets. Nonetheless, our single-stage approach shows clear strengths on all other metrics, particularly in aesthetics-related aspects: CE (+5.9%), CU (+9.4%), PC (+4.7%), and PQ (+7.5%), as well as in human evaluations, with Overall Quality (OVL.) increasing by 0.57 and Harmony (HAM.) by 1.04 on a five-point MOS scale. By directly modeling the joint distribution $P(\text{vocal}, \text{accompaniment})$, rather than optimizing $P(\text{vocal})$ and $P(\text{accompaniment} \mid \text{vocal})$ separately, the single-stage approach facilitates more effective global coordination and avoids error accumulation across stages. This is particularly beneficial in song generation, where the alignment and coherence between vocals and accompaniment are essential. We also showcase generation samples from both the multi-stage and single-stage models on our demo page [3], where the multi-stage outputs are often perceptibly off-beat. In addition to better quality, the single-stage model is also more efficient. On an A800 GPU, our mixed-pro model averages 18.04 seconds to generate a 30-second sample, compared to 42.85 seconds for the multi-stage baseline.

While SongGen shows some gaps when compared to Ground Truth and Suno, it is worth noting that we use only 2k hours of labeled data, sourced from publicly available datasets. Despite the limited data, SongGen achieves competitive performance. Figure 4 shows a mel-spectrogram of our generated songs, demonstrating that SongGen produces songs with various singing techniques like vibrato. Compared to Suno, a commercial product, SongGen outperforms in terms of text relevance and vocal control. Suno struggles to adhere to the highly detailed textual descriptions in MusicCaps (as shown by the REL. metric) and lacks voice cloning support, giving our model a distinct advantage in these aspects.

[3] https://liuzh-19.github.io/SongGen/

Table 1. Automatic evaluation of Text-to-Song generation. * denotes that we finetune Parlet-tts using our training data. The top two results, excluding the ground truth, are marked in **bold** and underlined, respectively.

	Metric / Model	Distrib. Match		Alignment				Aesthetics			
		FAD ↓	KL ↓	CLAP ↑	CLaMP3 ↑	PER ↓	SECS ↑	CE↑	CU↑	PC↑	PQ ↑
	Ground Truth	-	-	0.18	0.105	21.39	76.42	7.08	7.04	6.41	7.30
	Parler-tts*	4.13	1.00	0.19	0.074	58.61	64.37	5.96	6.62	5.49	6.82
	Multi-Stage	2.18	0.78	0.29	0.085	**38.80**	**74.04**	6.39	6.27	5.90	6.69
Mixed	Mixed	1.74	0.71	**0.35**	0.093	51.84	73.69	6.50	6.66	6.14	7.03
	Mixed pro (ours)	**1.71**	**0.69**	**0.35**	**0.094**	40.58	73.78	**6.77**	**6.86**	**6.18**	**7.19**
Dual-track	Parallel (standard)	2.45	0.75	0.33	0.087	48.40	72.27	6.21	6.40	5.68	6.88
	Parallel (V-A)	2.54	0.73	0.33	0.088	46.30	72.43	6.21	6.49	5.72	6.92
	Parallel (A-V)	2.31	0.72	0.34	0.089	47.00	72.50	6.26	6.47	5.80	6.92
	Interleaving (V-A)	1.96	0.71	0.34	0.092	41.82	73.12	6.52	6.52	5.97	7.03
	Interleaving (A-V) (ours)	1.87	**0.69**	**0.35**	0.093	39.46	73.16	6.67	6.72	6.11	7.12

Table 2. Humain evaluation of Text-to-Song generation. The overall first and second results are marked with **bold** and underline, respectively. The top results in both of our generation modes are highlighted in yellow.

	Model	OVL.↑	REL.↑	VQ.↑	HAM. ↑	SS. ↑
	Ground Truth	**4.57** ±0.04	**4.49** ±0.03	**4.49** ±0.05	**4.47** ±0.04	**4.58** ±0.03
	Suno	4.28±0.04	3.31±0.04	4.22±0.05	4.33±0.05	-
	Parler-tts*	2.58±0.06	2.13±0.05	2.28±0.03	2.35±0.04	-
	Multi-Stage	3.39±0.03	3.20±0.04	3.98±0.07	2.97±0.04	3.89±0.03
Mixed	Mixed	3.58±0.05	3.70±0.02	3.55±0.07	3.39±0.05	3.92±0.05
	Mixed pro	3.96 ±0.04	3.86±0.04	4.07±0.06	4.01±0.05	4.04±0.05
Dual-track	Parallel (std.)	3.19±0.04	3.27±0.06	3.36±0.04	2.98±0.05	3.44±0.03
	Parallel (V-A)	3.36±0.03	3.32±0.05	3.48±0.05	3.08±0.06	3.47±0.04
	Parallel (A-V)	3.40±0.03	3.33±0.04	3.51±0.04	3.21±0.05	3.51±0.05
	Inter. (V-A)	3.77±0.03	3.69±0.05	3.98±0.06	3.65±0.04	3.88±0.04
	Inter. (A-V)	3.95±0.03	3.87±0.06	4.15±0.05	3.82±0.03	3.93±0.04

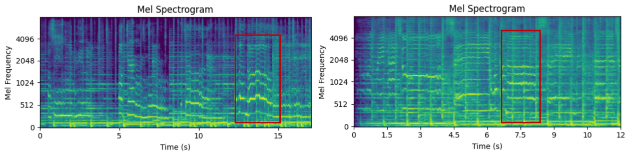

Figure 4. Mel-spectrogram visualization of our generated song featuring various singing techniques.

Mixed Mode and Dual-Track Mode. We further analyze the performance of the mixed mode and dual-track mode of our framework. In mixed mode generation, the ”Mixed Pro” approach outperforms the basic ”Mixed” model across all metrics, particularly in vocal quality (as indicated by the PER and VQ.). It indicates that by incorporating an auxiliary vocal token prediction target, the learning biases in mixed mode are effectively mitigated.

In dual-track mode, the “Interleaving (A-V)” pattern obtains the best performance. Although the parallel pattern is more computationally efficient, its performance lags behind the interleaving pattern. This is likely because, in parallel mode, each hidden state mixes vocals and accompaniment, making separation difficult with only two linear heads. Interestingly, regardless of the pattern (parallel or interleaving), placing the accompaniment before the vocals leads to better results than the reverse order.

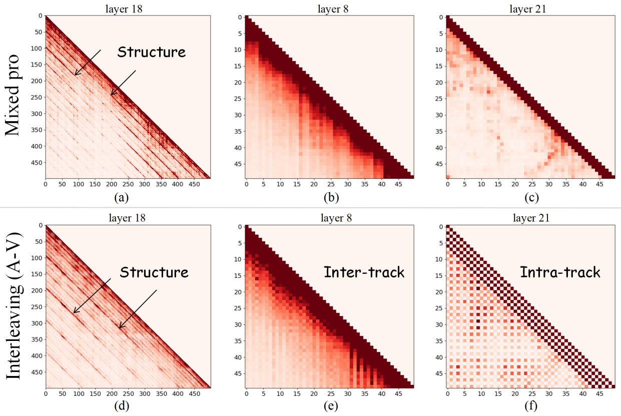

Figure 5. Visualization of decoder attention.

Compared to “Mixed pro”,“Interleaving (A-V)” shows competitive performance, with only slightly worse result in FAD. Further comparison reveals that “Interleaving (A-V)” achieves better vocal quality (VQ.), but its harmony (HAM.) is slightly inferior to that of the “Mixed pro”. This highlights the distinct advantages and challenges of each generation mode. We further visualize the attention scores in the decoder to explore the internal mechanisms of the transformer in both modes. Figures 5 (a),(b), and (c) show self-attention over 500 steps in layer 18 for the “mixed pro”, and over 50 steps in layers 8 and 21. Figures 5 (a),(b), and (c) present the same for the “interleaving (A-V)” pattern. From (a) and (d), we observe evenly spaced parallel lines along the diagonal. Since songs typically have repetitive structures, this atten-

Table 3. Text-to-Song results without voice input.

Model	FAD ↓	OVL.↑	REL.↑	VQ.↑	HAM. ↑
Mixed pro	1.96	3.72±0.03	3.48±0.04	3.88±0.06	3.87±0.05
Inter.(A-V)	2.21	3.70±0.04	3.47±0.06	3.91±0.04	3.83±0.04

Table 4. Ablation results on training scheme. HQFT is short for High-Quality Finetuning and CL stands for curriculum learning.

Model	FAD ↓	KL ↓	CLAP ↑	PER ↓	SECS ↑
w/o HQFT	2.01	0.72	0.32	43.68	72.83
w/o CL	2.35	0.73	0.33	55.71	72.81
ours	**1.71**	**0.69**	**0.35**	**40.58**	**73.78**

tion pattern suggests that our model has effectively learned the underlying structure of music. Interestingly, in (f), the attention follows a checkerboard pattern, where attention scores for odd steps are strong with other odd steps and similarly, for even steps. This indicates that in the "interleaving (A-V)" mode, higher layers focus more on learning intra-track relationships, while lower layers (shown in (c)) capture inter-track interactions.

Without a reference voice. We explore the song generation capability of SongGen without a reference voice. Table 3 shows that performance declines slightly. However, the listening test results demonstrate that the model continues to produce enjoyable songs with coherent vocals.

4.3. Ablation Studies

In this section, we conduct extensive ablation studies. Since both mode are based on a unified framework, we present results from the mixed mode setting due to space limitations.

Effect of training strategy. In Table 4, we evaluate the effectiveness of our High-Quality Finetuning (HQFT) and curriculum learning (CL) strategy for codebook loss weights. HQFT improves all metrics, confirming the effectiveness of our quality filtering criteria. Compared to the "w/o CL" variant, where each codebook's loss weight is fixed and equal, our CL strategy improves performance. This demonstrates that prioritizing the most important tasks first and then progressively refining the details is effective.

Effect of Lyrics Module Design. We further investigate the impact of different lyric integration methods, including the choice of tokenizer (VoiceBPE vs. T5), the use of a lyrics encoder, and the integration approach (pre-pending vs. cross-attention). Figure 5 shows the results after Step 1 training for each variant. Our design (VoiceBPE, w/ lyrics encoder, cross-attention) achieves the best results across all metrics, validating the effectiveness. Unlike most TTS works, which prepend transcripts before audio tokens, we find that the cross-attention approach is more effective and stable. This may be because cross-attention allows the decoder to focus solely on generating the audio modality. Additionally, phoneme-like tokenizer (VoiceBPE) is more suitable than the word-level tokenizer (T5) for song lyric tokenization. Under this mechanism, the lyrics encoder can capture the relationships between lyric tokens, learning pronunciation patterns from different token combinations, and thus alleviating the burden of modality alignment on the decoder.

Table 5. Ablation results on different lyric integration methods.

Tokenizer	w/ Lyrics Encoder	prepend / cross	FAD ↓	PER ↓	SECS ↑
VoiceBPE	✗	prepend	3.41	62.38	69.09
VoiceBPE	✓	prepend	3.56	56.21	70.70
VoiceBPE	✗	cross	1.95	61.81	72.59
T5	✓	cross	1.88	55.27	73.67
VoiceBPE	✓	cross	**1.73**	**43.34**	**73.59**

5. Limitations and Future Work

We acknowledge the limitations of our proposed SongGen model. Due to the scarcity of open-source song data, the current model can only generate songs up to 30 seconds in length, which is insufficient for producing songs with complete structures. Additionally, the current audio codec, X-Codec, operates at a sampling rate of 16kHz. To improve fidelity, our future work will involve training a renderer to upsample the audio for higher quality output.

6. Conclusion

In this paper, we introduced SongGen, a fully open-source, single-stage auto-regressive transformer for text-to-song generation. Operating within a unified framework, we devised a variety of token patterns. These patterns endow SongGen with the ability to support two distinct generation modes: the mixed mode and the dual-track mode. Experimental outcomes convincingly demonstrate the efficacy of our token pattern design. Moreover, they showcase the strong song generation capabilities of SongGen in both the mixed mode and the dual-track mode.

Acknowledgments

This work was supported by National Key R&D Program of China 2022ZD0161600, Shanghai Artificial lntelligence Laboratory, the Centre for Perceptual and Interactive Intelligence (CPII) Ltd under the Innovation and Technology Commission (ITC)'s InnoHK. Dahua Lin is a PI of CPII under the InnoHK.

Impact Statement

The proposed work, SongGen, a controllable text-to-song generation model, has the potential to impact various aspects of society. On the positive side, SongGen enables both content creators and novices to effortlessly express their creativity with a low entry barrier, while also streamlining the workflow for experienced music producers.

However, since SongGen autonomously generates songs and supports voice cloning, there are risks of copyright infringement, intellectual property misuse, and the creation of deepfake audio. Proper constraints are needed to ensure the model is not misused in illegal or unethical ways.

In conclusion, while SongGen presents exciting possibilities for the music industry and creative expression, its development should be accompanied by careful consideration of its ethical and societal implications

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Agostinelli, A., Denk, T. I., Borsos, Z., Engel, J., Verzetti, M., Caillon, A., Huang, Q., Jansen, A., Roberts, A., Tagliasacchi, M., et al. Musiclm: Generating music from text. *arXiv preprint arXiv:2301.11325*, 2023.

Bai, Y., Chen, H., Chen, J., Chen, Z., Deng, Y., Dong, X., Hantrakul, L., Hao, W., Huang, Q., Huang, Z., Jia, D., La, F., Le, D., Li, B., Li, C., Li, H., Li, X., Liu, S., Lu, W.-T., Lu, Y., Shaw, A., Spijkervet, J., Sun, Y., Wang, B., Wang, J.-C., Wang, Y., Wang, Y., Xu, L., Yang, Y., Yao, C., Zhang, S., Zhang, Y., Zhang, Y., Zhao, H., Zhao, Z., Zhong, D., Zhou, S., and Zou, P. Seed-music: A unified framework for high quality and controlled music generation, 2024. URL https://arxiv.org/abs/2409.09214.

Bertin-Mahieux, T., Ellis, D. P., Whitman, B., and Lamere, P. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

Bogdanov, D., Won, M., Tovstogan, P., Porter, A., and Serra, X. The mtg-jamendo dataset for automatic music tagging. In *Machine Learning for Music Discovery Workshop, International Conference on Machine Learning (ICML 2019)*, Long Beach, CA, United States, 2019. URL http://hdl.handle.net/10230/42015.

Casanova, E., Davis, K., Gölge, E., Göknar, G., Gulea, I., Hart, L., Aljafari, A., Meyer, J., Morais, R., Olayemi, S., and Weber, J. Xtts: a massively multilingual zero-shot text-to-speech model, 2024. URL https://arxiv.org/abs/2406.04904.

Chen, K., Wu, Y., Liu, H., Nezhurina, M., Berg-Kirkpatrick, T., and Dubnov, S. Musicldm: Enhancing novelty in text-to-music generation using beat-synchronous mixup strategies. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1206–1210. IEEE, 2024a.

Chen, S., Liu, S., Zhou, L., Liu, Y., Tan, X., Li, J., Zhao, S., Qian, Y., and Wei, F. Vall-e 2: Neural codec language models are human parity zero-shot text to speech synthesizers, 2024b. URL https://arxiv.org/abs/2406.05370.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. Scaling instruction-finetuned language models, 2022. URL https://arxiv.org/abs/2210.11416.

Copet, J., Kreuk, F., Gat, I., Remez, T., Kant, D., Synnaeve, G., Adi, Y., and Défossez, A. Simple and controllable music generation. *Advances in Neural Information Processing Systems*, 36, 2024.

Defferrard, M., Mohanty, S. P., Carroll, S. F., and Salathé, M. Learning to recognize musical genre from audio. In *The 2018 Web Conference Companion*. ACM Press, 2018. ISBN 9781450356404. doi: 10.1145/3184558.3192310. URL https://arxiv.org/abs/1803.05337.

Défossez, A., Copet, J., Synnaeve, G., and Adi, Y. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., and Sutskever, I. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

Doh, S., Choi, K., Lee, J., and Nam, J. Lp-musiccaps: Llm-based pseudo music captioning. In *Ismir 2023 Hybrid Conference*, 2023.

Evans, Z., Parker, J. D., Carr, C., Zukowski, Z., Taylor, J., and Pons, J. Stable audio open. *arXiv preprint arXiv:2407.14358*, 2024.

Forsgren, S. and Martiros, H. Riffusion-stable diffusion for real-time music generation. *URL https://riffusion. com*, 2022.

Gandhi, S., von Platen, P., and Rush, A. M. Distil-whisper: Robust knowledge distillation via large-scale pseudo labelling, 2023.

Gao, Z., Li, Z., Wang, J., Luo, H., Shi, X., Chen, M., Li, Y., Zuo, L., Du, Z., Xiao, Z., and Zhang, S. Funasr: A fundamental end-to-end speech recognition toolkit. In *INTERSPEECH*, 2023.

Gemmeke, J. F., Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 776–780. IEEE, 2017.

Hershey, S., Chaudhuri, S., Ellis, D. P. W., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., Slaney, M., Weiss, R. J., and Wilson, K. Cnn architectures for large-scale audio classification, 2017. URL https://arxiv.org/abs/1609.09430.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Hong, Z., Huang, R., Cheng, X., Wang, Y., Li, R., You, F., Zhao, Z., and Zhang, Z. Text-to-song: Towards controllable music generation incorporating vocals and accompaniment, 2024. URL https://arxiv.org/abs/2404.09313.

Huang, C.-Z. A., Vaswani, A., Uszkoreit, J., Simon, I., Hawthorne, C., Shazeer, N., Dai, A. M., Hoffman, M. D., Dinculescu, M., and Eck, D. Music transformer: Generating music with long-term structure. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJe4ShAcF7.

Huang, Q., Park, D. S., Wang, T., Denk, T. I., Ly, A., Chen, N., Zhang, Z., Zhang, Z., Yu, J., Frank, C., et al. Noise2music: Text-conditioned music generation with diffusion models. *arXiv preprint arXiv:2302.03917*, 2023.

Ji, S., Luo, J., and Yang, X. A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions. *arXiv preprint arXiv:2011.06801*, 2020.

Kilgour, K., Zuluaga, M., Roblek, D., and Sharifi, M. Fréchet audio distance: A metric for evaluating music enhancement algorithms, 2019. URL https://arxiv.org/abs/1812.08466.

Kingma, D., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Kumar, R., Seetharaman, P., Luebs, A., Kumar, I., and Kumar, K. High-fidelity audio compression with improved rvqgan. *Advances in Neural Information Processing Systems*, 36, 2024.

Lei, S., Zhou, Y., Tang, B., Lam, M. W. Y., Liu, F., Liu, H., Wu, J., Kang, S., Wu, Z., and Meng, H. Songcreator: Lyrics-based universal song generation, 2024. URL https://arxiv.org/abs/2409.06029.

Li, R., Hong, Z., Wang, Y., Zhang, L., Huang, R., Zheng, S., and Zhao, Z. Accompanied singing voice synthesis with fully text-controlled melody, 2024a. URL https://arxiv.org/abs/2407.02049.

Li, Y., Yuan, R., Zhang, G., Ma, Y., Chen, X., Yin, H., Xiao, C., Lin, C., Ragni, A., Benetos, E., Gyenge, N., Dannenberg, R., Liu, R., Chen, W., Xia, G., Shi, Y., Huang, W., Wang, Z., Guo, Y., and Fu, J. Mert: Acoustic music understanding model with large-scale self-supervised training, 2024b. URL https://arxiv.org/abs/2306.00107.

Liu, H., Yuan, Y., Liu, X., Mei, X., Kong, Q., Tian, Q., Wang, Y., Wang, W., Wang, Y., and Plumbley, M. D. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Lyth, D. and King, S. Natural language guidance of high-fidelity text-to-speech with synthetic annotations. *arXiv preprint arXiv:2402.01912*, 2024.

Ning, Z., Wang, S., Jiang, Y., Yao, J., He, L., Pan, S., Ding, J., and Xie, L. Drop the beat! freestyler for accompaniment conditioned rapping voice generation, 2024. URL https://arxiv.org/abs/2408.15474.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. Robust speech recognition via large-scale weak supervision, 2022.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21 (140):1–67, 2020.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Rouard, S., Massa, F., and Défossez, A. Hybrid transformers for music source separation. In *ICASSP 23*, 2023.

Rouard, S., Roman, R. S., Adi, Y., and Roebel, A. Musicgen-stem: Multi-stem music generation and edition through autoregressive modeling, 2025. URL https://arxiv.org/abs/2501.01757.

Schneider, F., Kamal, O., Jin, Z., and Schölkopf, B. Mo\^ usai: Text-to-music generation with long-context latent diffusion. *arXiv preprint arXiv:2301.11757*, 2023.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Tjandra, A., Wu, Y.-C., Guo, B., Hoffman, J., Ellis, B., Vyas, A., Shi, B., Chen, S., Le, M., Zacharov, N., Wood, C., Lee, A., and Hsu, W.-N. Meta audiobox aesthetics: Unified automatic quality assessment for speech, music, and sound. 2025. URL https://arxiv.org/abs/2502.05139.

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

Wu, S., Guo, Z., Yuan, R., Jiang, J., Doh, S., Xia, G., Nam, J., Li, X., Yu, F., and Sun, M. Clamp 3: Universal music information retrieval across unaligned modalities and unseen languages, 2025. URL https://arxiv.org/abs/2502.10362.

Wu*, Y., Chen*, K., Zhang*, T., Hui*, Y., Berg-Kirkpatrick, T., and Dubnov, S. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, 2023.

Yang, D., Liu, S., Huang, R., Tian, J., Weng, C., and Zou, Y. Hifi-codec: Group-residual vector quantization for high fidelity audio codec. *arXiv preprint arXiv:2305.02765*, 2023.

Ye, Z., Sun, P., Lei, J., Lin, H., Tan, X., Dai, Z., Kong, Q., Chen, J., Pan, J., Liu, Q., Guo, Y., and Xue, W. Codec does matter: Exploring the semantic shortcoming of codec for audio language model. *arXiv preprint arXiv:2408.17175*, 2024.

Yuan, R., Ma, Y., Li, Y., Zhang, G., Chen, X., Yin, H., Zhuo, L., Liu, Y., Huang, J., Tian, Z., Deng, B., Wang, N., Lin, C., Benetos, E., Ragni, A., Gyenge, N., Dannenberg, R., Chen, W., Xia, G., Xue, W., Liu, S., Wang, S., Liu, R., Guo, Y., and Fu, J. Marble: Music audio representation benchmark for universal evaluation, 2023. URL https://arxiv.org/abs/2306.10548.

Yuan, R., Lin, H., Guo, S., Zhang, G., Pan, J., Zang, Y., Liu, H., Liang, Y., Ma, W., Du, X., Du, X., Ye, Z., Zheng, T., Jiang, Z., Ma, Y., Liu, M., Tian, Z., Zhou, Z., Xue, L., Qu, X., Li, Y., Wu, S., Shen, T., Ma, Z., Zhan, J., Wang, C., Wang, Y., Chi, X., Zhang, X., Yang, Z., Wang, X., Liu, S., Mei, L., Li, P., Wang, J., Yu, J., Pang, G., Li, X., Wang, Z., Zhou, X., Yu, L., Benetos, E., Chen, Y., Lin, C., Chen, X., Xia, G., Zhang, Z., Zhang, C., Chen, W., Zhou, X., Qiu, X., Dannenberg, R., Liu, J., Yang, J., Huang, W., Xue, W., Tan, X., and Guo, Y. Yue: Scaling open foundation models for long-form music generation, 2025. URL https://arxiv.org/abs/2503.08638.

Zeghidour, N., Luebs, A., Omran, A., Skoglund, J., and Tagliasacchi, M. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

Zhang, L., Rao, A., and Agrawala, M. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

A. Implementation and Training Details

In SongGen, the lyrics encoder is a 6-layer transformer with a hidden size of 1024. The transformer decoder, consisting of 24 layers with 1024 hidden size, includes both causal self-attention and cross-attention blocks in each layer. In "Mixed Pro" Mode, the vocal loss weight λ is set to 0.1. The model is trained for approximately 400K steps using 16 Nvidia A100 (80GB) GPUs, with a batch size of 16 per GPU. For optimization, we employ the AdamW optimizer (Loshchilov & Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and a weight decay of 10^{-4}. During training step 1, the learning rate is set to 10^{-4}, while for the subsequent fine-tuning steps, the learning rate is reduced to 5×10^{-5}. A cosine learning rate schedule is applied for all traning steps. The resource allocation and training strategy for the multi-stage baseline are consistent with those used for SongGen mixed training Step 1. Since voice-free support is not directly related to the core comparison between the single-stage and multi-stage designs, we omit this part of the multi-stage model. Specifically, the two models in the multi-stage pipeline are trained separately for approximately 200K steps each. We observe that the loss begins to plateau around 60K steps for both models. To facilitate reproducibility, we make our training configurations publicly available.

B. Details in Evaluations

For evaluation, we select 326 samples from the MusicCaps (Agostinelli et al., 2023) benchmark, with no overlap with the training set. MusicCaps test set contains 2.8K samples, with captions written by expert musicians. However, many of the samples are instrumental music, sound effects, or speech. We filter the English-voiced song samples from MusicCaps test set using our automated data preprocessing pipeline, resulting a final set of 326 samples. Note that the evaluation set was selected impartially, with no intention to influence fairness.

For objective metrics, all samples are normalized at-14dB LUFS for fairness. Frechet Audio Distance (FAD) (Kilgour et al., 2019) evaluates the fidelity of generated songs by calculating the distribution distance between features of the target and generated audio, extracted from the VGGish (Hershey et al., 2017) model. Kullback-Leibler Divergence (KL) measures the similarity between the generated and target audio with the label calculated by the audio tagging model. A lower KL suggests that the generated music shares similar concepts with the reference. FAD and KL are calculated using `audioldm_eval` [4]. CLAP Score evaluates the alignment between generated audio and the given text prompt using the official CLAP model (Wu* et al., 2023), implemented via the `stable-audio-metrics` [5]. However, we observe that the CLAP Score for MusicCaps test set is unexpectedly low. We supplement our evaluation with the CLaMP3 Score (Wu et al., 2025), a more recent model that provides a more robust measure of text–song alignment. Phoneme Error Rate (PER) assesses the adherence of the generated audio to the provided lyrics by transcribing the audio using Distill Whisper(Gandhi et al., 2023) and computing the phoneme error rate against the reference lyrics. However, PER is not an ideal measure of vocal quality, as current ASR models struggle with sung vocals. Speaker Embedding Cosine Similarity (SECS) assesses the similarity of speaker identity using the Resemblyzer[6] speaker encoder to compute the SECS between reference 3-second vocal clips and generated audio. We further introduce recently proposed content-based aesthetics metrics (Tjandra et al., 2025), covering Content Enjoyment (CE), Content Usefulness (CU), Production Complexity (PC), and Production Quality (PQ).

For the subjective evaluations, we randomly select 36 audio samples generated by our models, and each sample is evaluated by 20 listeners. We conduct the commonly used MOS (Mean Opinion Score) tests across five aspects. The rating scale ranges from 1 to 5, with higher scores indicating better performance. For the Overall Quality (OVL.) evaluation, we instruct the raters to focus on musicality and naturalness, while ignoring style differences. For the Relevance to Text Description (REL.) evaluation, we ask the raters to score based on the proportion of key points from the text description that are reflected in the generated song. For the Vocal Quality (VQ.) evaluation, we emphasize the importance of clarity, lyric accuracy, and the naturalness and coherence of the vocals in the ratings. For Harmony (HAM.), we ask the raters to pay particular attention to the temporal correspondence between the accompaniment and the vocals. For Speaker Similarity, we ask the raters to focus on the similarity of the speaker's identity (timbre) to the reference, ignoring differences in content. A small subset of the samples used in the test is available on our project page `https://liuzh-19.github.io/SongGen/`.

[4] `https://github.com/haoheliu/audioldm_eval`

[5] `https://github.com/Stability-AI/stable-audio-metrics`

[6] `https://github.com/resemble-ai/Resemblyzer`

Table 6. Ablation results of different neural audio codecs.

Model	FAD ↓	KL ↓	CLAP ↑	PER ↓	SECS ↑
Encodec	10.84	0.99	0.19	60.67	71.36
DAC	4.36	0.86	0.24	68.64	71.66
X-Codec (ours)	**1.73**	**0.70**	**0.33**	**43.34**	**73.59**

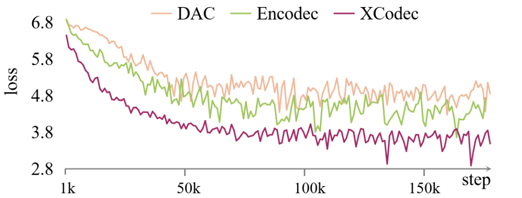

Figure 6. Training loss curves of different audio codecs

C. The Impact of Different Audio Codecs.

We compare the performance of three different codecs: XCodec, Encodec (24kHz) (Défossez et al., 2022), and DAC (44.1kHz) (Kumar et al., 2024). Table 6 shows the results after training Step 1. X-Codec surpasses both Encodec and DAC on all metrics. Additionally, the loss curves in Figure 6 demonstrate that X-Codec exhibits more stable training and faster convergence. Although Encodec and DAC have been widely adopted in prior audio generation systems across domains such as speech (Lyth & King, 2024) and instrumental music (Copet et al., 2024), song generation presents a substantially higher level of semantic complexity. While Encodec and DAC indeed yield better perceptual quality for audio reconstruction, we observed that in song generation, both codecs resulted in higher rates of invalid outputs, such as failure to follow lyrics, or producing noise and silence. In contrast, X-Codec consistently demonstrated more stable training, faster convergence, and higher success rates in generating coherent vocals.

We attribute this performance to several factors. First, we adopt the publicly released `xcodec_hubert_general_audio` checkpoint, trained on a large-scale (200k-hour) private dataset with a distribution similar to AudioSet (Gemmeke et al., 2017). We speculate that its exposure to large amounts of music data during pretraining contributes to its superior performance in our task. Second, as emphasized in the X-Codec paper (Ye et al., 2024), the incorporation of not only acoustic but also semantic features from self-supervised learning representations might also contribute to the performance. Despite X-Codec operating at a relatively low sampling rate of 16 kHz, we selected it as the most suitable option available at the time. To date, high-fidelity, song-specific neural codecs tailored for generative modeling remain an open challenge in the research community.