Versatile Framework for Song Generation with Prompt-based Control

Anonymous ACL submission

Abstract

Song generation focuses on producing controllable high-quality songs based on various prompts. However, existing methods struggle to generate vocals and accompaniments with prompt-based control and proper alignment. Additionally, they fall short in supporting various tasks. To address these challenges, we introduce VersBand, a multi-task song generation framework for synthesizing high-quality, aligned songs with prompt-based control. VersBand comprises these primary models: 1) VocalBand, a decoupled model, leverages the flow-matching method for generating singing styles, pitches, and mel-spectrograms, allowing fast, high-quality vocal generation with style control. 2) AccompBand, a flow-based transformer model, incorporates the Band-MOE, selecting suitable experts for enhanced quality, alignment, and control. This model allows for generating controllable, high-quality accompaniments aligned with vocals. 3) Two generation models, LyricBand for lyrics and MelodyBand for melodies, contribute to the comprehensive multi-task song generation system, allowing for extensive control based on multiple prompts. Experimental results demonstrate that VersBand performs better over baseline models across multiple song generation tasks using objective and subjective metrics. Audio samples are available at https://VersBand.github.io.

1 Introduction

Song generation focuses on producing complete musical pieces based on text prompts (about lyrics, melodies, singing, and music styles) and optional audio prompts. Unlike singing voice synthesis (SVS) (Shi et al., 2022; Cho et al., 2022), which focuses on the singing component, or music generation (Dong et al., 2018; Agostinelli et al., 2023) for synthesizing only instrumental tracks, song generation involves synthesizing both high-quality vocals and accompaniments with high-level prompt-based control and proper alignment (Li et al., 2024a).

Despite significant advancements in SVS and music domains, generating high-quality, controllable, aligned songs remains challenging. Song generation aims to enable controllable musical experiences, with broad applications ranging from entertainment videos to professional composition. As shown in Figure 1, song generation models can leverage different prompts for multiple tasks. **Text prompts** allow for control over tasks, lyrics, melody, singing styles (like singing methods, emotion, and techniques), and music styles (like genre, tone, and instrumentation), while **audio prompts** enable users to input their voice or preferred music for customization. However, the few existing song generation models (Zhiqing et al., 2024) lack mechanisms to properly align vocals with accompaniments and fail to achieve effective control.

Currently, song generation encounters three major challenges: 1) **Limitations in high-quality vocal generation with style control.** For singing style control, StyleSinger (Zhang et al., 2024a) conducts style transfer, while PromptSinger (Wang et al., 2024) achieves singer identity control. However, existing models have yet to generate pleasing vocals with high-level style control (like singing methods, emotion, and techniques) by text prompts, and customization with audio prompts. 2) **Difficulties in controllable and aligned accompaniment generation.** Existing music generation models (Dong et al., 2018) and text-to-song models (Zhiqing et al., 2024) lack mechanisms for style control and aligning vocals with accompaniments in rhythm, melody, and beats. Generating controllable (like genre, tone, and instrumentation) and aligned accompaniments remains challenging. 3) **Challenges in multi-task song generation based on various prompts.** The limited existing song generation methods (Li et al., 2024a) do not support diverse related tasks. This reliance on constrained inputs leads to a suboptimal user experience and restricts the models' ability to customize songs.

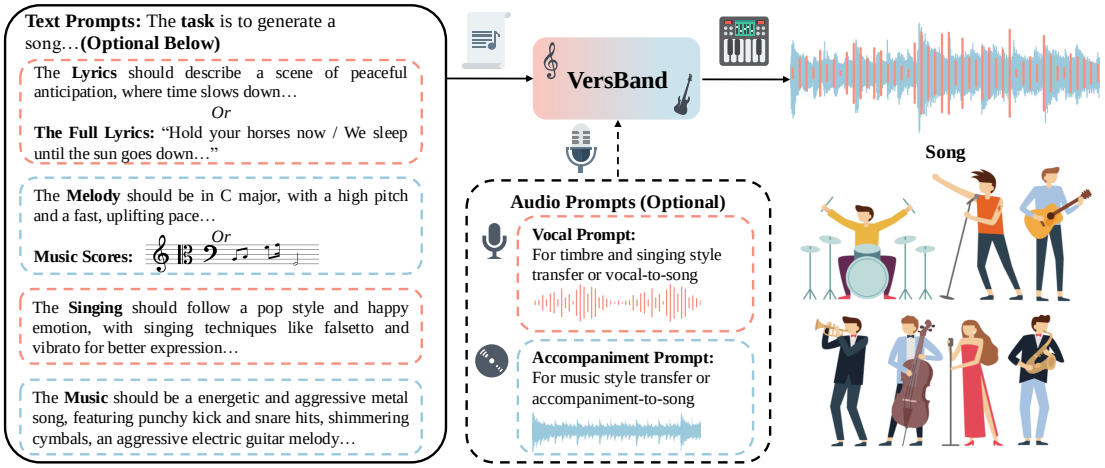


Figure 1: Overview of VersBand, which generates complete songs like a versatile band. The dashed lines indicate optional inputs. At a minimum, users can just input "The task is to generate a song."

To address these challenges, we introduce VersBand, a multi-task song generation framework for synthesizing high-quality, aligned songs with prompt-based control. Following the human perception that accompaniment complements vocal melody with complex harmonic and rhythmic structure (Zhiqing et al., 2024), we generate them separately. To achieve fast and high-quality vocal generation with control, we design a decoupled model VocalBand, predicting singing styles, pitches, and mel-spectrograms based on the flow-matching method. Based on the complex nature of music, we introduce a flow-based transformer model AccompBand to generate high-fidelity, controllable, aligned accompaniments. We design Band-MOE (Mixture of Experts), selecting suitable experts for enhanced quality, alignment, and control. Additionally, we add two generation models, LyricBand for lyrics and MelodyBand for melodies, contributing to the comprehensive multi-task song generation system. Our experiments on open-source and web-crawled bilingual song datasets show VersBand can generate high-quality songs with control, outperforming baseline models in multiple song generation tasks. The main contributions of our work are summarized as follows:

- We propose VersBand, a multi-task song generation approach for generating high-quality, aligned songs with prompt-based control.
- We design a decoupled model VocalBand, which leverages the flow-matching method to generate singing styles, pitches, and mel-spectrograms, enabling fast and high-quality vocal synthesis with high-level style control.
- We introduce a flow-based transformer model AccompBand to generate high-quality, controllable, aligned accompaniments, with the Band-MOE, selecting suitable experts for enhanced quality, alignment, and control.
- Experimental results demonstrate that VersBand achieves superior objective and subjective evaluations compared to baseline models across multiple song generation tasks.

2 Background

2.1 Singing Voice Synthesis

Singing Voice Synthesis (SVS) rapidly advances for generating singing voices from given lyrics and music scores. Choi and Nam (2022) presents a melody-unsupervised model, eliminating the need for temporal alignment. VISinger 2 (Zhang et al., 2022b) employs digital signal processing techniques to enhance fidelity, while Kim et al. (2024) uses adversarial multi-task learning to improve the singing naturalness. StyleSinger (Zhang et al., 2024a) facilitates style transfer by extracting styles via a residual quantization method. Additionally, PromptSinger (Wang et al., 2024) attempts to control speaker identity based on text descriptions. GTSinger (Zhang et al., 2024b) releases a dataset with multiple style annotations. Despite these advancements, these approaches can not generate aligned accompaniment. Recently, Melodist (Zhiqing et al., 2024) has introduced a text-to-song model that sequentially generates vocals and accompaniments using auto-regressive transformers. However, achieving high-quality vocal generation with high-level control remains challenging.

2.2 Accompaniment Generation

Research in accompaniment usually focuses on musical symbolic tokens. MuseGAN (Dong et al., 2018) employs a GAN-based approach to generate symbolic music. SongMASS (Sheng et al., 2021) uses transformer models to generate lyrics or melodies conditioned on each other. MusicLM (Agostinelli et al., 2023) leverages joint textual-music representations from MuLan (Huang et al., 2022) to generate semantic and acoustic tokens using transformer decoders. MusicLDM (Chen et al., 2024) incorporates beat-tracking information to address potential plagiarism concerns in music generation. Additionally, SingSong (Donahue et al., 2023) introduces a model that generates background music to complement the provided vocals. SongCreator (Lei et al., 2024) can create songs based on lyrics and audio prompts. Recently, MelodyLM (Li et al., 2024a) has employed transformer and diffusion models for decoupled song generation. Nevertheless, challenges remain in generating high-quality music with effective control based on multiple prompts. Existing methods also lack mechanisms for alignment with vocals and support for multiple song generation tasks.

3 Method

This section introduces VersBand. We design two distinct models, VocalBand for vocals and AccompBand for accompaniments, tailored to their unique characteristics. Additionally, we include LyricBand for lyrics and MelodyBand for melodies, composing a multi-task song generation system.

3.1 Multi-Task Song Generation

As shown in Figure 2 (a), VersBand handles multi-task song generation based on text and audio prompts. We employ a text encoder to generate text tokens z_p. When lyrics or music scores are not provided, LyricBand and MelodyBand predict phonemes p and notes n (pitch and duration) as target contents. Next, in Figure 2 (b), to achieve fast and high-quality vocal generation with granular control, we introduce VocalBand, which decouples the content z_c, timbre z_t, and style z_s. Through the Flow-based Pitch Predictor, Mel Decoder, and pre-trained vocoder, the target vocal y_v is synthesized. Then, in Figure 2 (c), for the complex nature of accompaniment, we design AccompBand to achieve superior quality, alignment, and control. AccompBand uses two encoders to extract embeddings z_v from y_v and x from ground truth (GT) accompaniment $\hat{y_a}$ during training. z_v and noise-injected x_t are processed by Band Transformer Blocks with Band-MOE, which selects suitable experts by z_v, z_p, and time step t for enhanced quality, alignment, and control. During inference, the ordinary differential equation (ODE) solver, accomp decoder, and vocoder generate the target accompaniment y_a. y_v and y_a are combined to the final target song y.

3.2 VocalBand

Decomposition As shown in Figure 2(b), for more personalized and fine-grained control, we disentangle target vocal y_v into content z_c, style z_s (e.g., singing methods, emotion, techniques, pronunciation, articulation skills), and timbre z_t. For z_c, phonemes p and notes n (note pitch and duration) are encoded by a phoneme encoder and a note encoder. Given a vocal prompt $\tilde{y_v}$, the timbre and personalized styles (like pronunciation and articulation skills) should remain consistent. We pass $\tilde{y_v}$ through a timbre encoder to obtain $\tilde{z_t}$, while $z_t = \tilde{z_t}$. Next, the residual style encoder employs an RQ model (Lee et al., 2022a) to extract phoneme-level prompt style $\tilde{z_s}$. This serves as an information bottleneck to filter out non-style information (Zhang et al., 2024a), ensuring effective decomposition. The Flow-based Style Predictor uses z_c, z_t, $\tilde{z_s}$, and text tokens z_p to predict z_s, learning both personalized styles of $\tilde{z_s}$ and style control information in z_p (like singing methods). For more details, please refer to Appendix C.2.

Flow-based Style Predictor Singing styles typically exhibit continuous and complex dynamics, involving intricate variations. The flow-matching model (Liu et al., 2022) is suitable for generating styles with finer-grained control by modeling styles as a smooth transformation, effectively balancing multiple control inputs, enabling a fast and stable generation of natural and consistent styles.

As shown in Figure 3 (a), we design the Flow-based Style Predictor using content z_c, timbre z_t, prompt style $\tilde{z_s}$, and text tokens z_p to predict the target style z_s. With input z_c and z_t, we employ a style alignment model with the Scaled Dot-Product Attention mechanism (Vaswani et al., 2017) to align style control information from z_p (e.g., singing methods, emotions, techniques) with contents. The fused condition c serves as the condition for an ODE solver, which transforms Gaussian noise ϵ into z_s along a smooth probability path $p_t(z_{st})$.

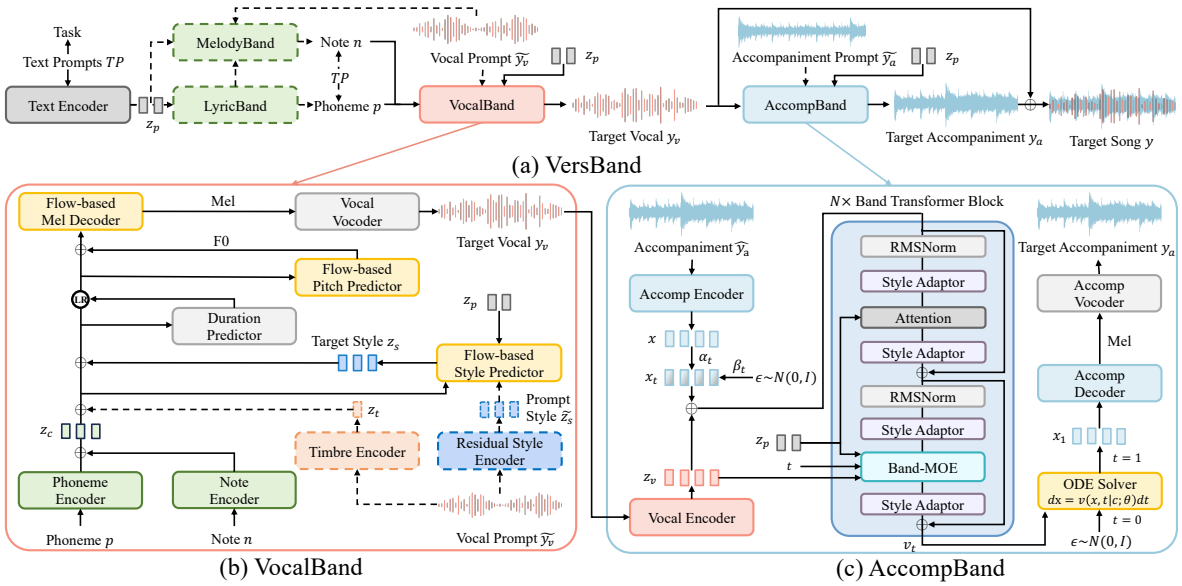


Figure 2: The overall architecture of VersBand. Vocal and accompaniment are generated by VocalBand and AccompBand separately. Dashed lines represent optional processes, while LR stands for length regulator.

z_{st} is obtained by linear interpolation at time t between ϵ and z_s, which is extracted from the GT vocal by the residual style encoder, thus the target vector field $u(z_{st}, t) = z_s - \epsilon$. We concatenate $\tilde{z}_s$ with z_{st} to allow z_s to learn personalized styles (e.g., pronunciation, articulation skills). The learned vector field $v_t(z_{st}, t|c; \theta)$, predicted by a vector field estimator at each time t, ensures smooth interpolation between the initial noise and output, guided by the flow-matching objective, minimizing the distance of learned and true vector fields:

$$\mathcal{L}_{style} = \mathbb{E}_{t, p_t(z_{st})} \left\| v_t(z_{st}, t|c; \theta) - (z_s - \epsilon) \right\|^2. \quad (1)$$

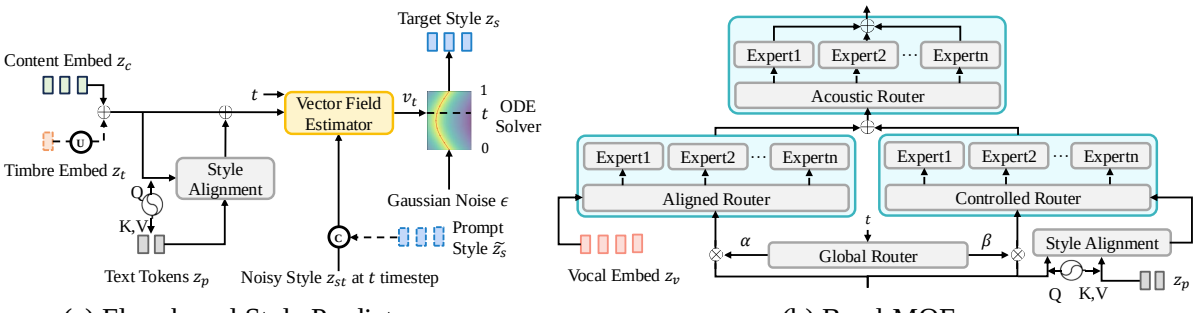
where $p_t(z_{st})$ represents the distribution of z_{st} at time t. This method ensures the fast and controlled generation of target style z_s, learning both personalized styles consistent with $\tilde{z}_s$ and aligned style control information from z_p. Notably, $\tilde{z}_s$ and z_p can be input individually for full style control. For more details, please refer to Appendix A and C.6.

Flow-based Pitch Predictor and Mel Decoder Traditional pitch predictors and mel decoders struggle to capture the dynamic and complex variations in singing voices. Thus, we propose the Flow-based Pitch Predictor and Mel Decoder, which use content z_c, timbre z_t, and style z_s to quickly and robustly generate high-quality F0 and mel-spectrograms with a similar architecture to Flow-based Style Predictor. Our pitch loss $\mathcal{L}_{pitch}$ and mel loss $\mathcal{L}_{mel}$ are analogous to $\mathcal{L}_{style}$ in Equation 1. For more details, please refer to Appendix C.8.

3.3 AccompBand

Band Transformer Block Accompaniment generation is highly complex due to the intricate interplay of various instruments and alignment with vocals, especially for long-sequence generation. Flow matching enables smooth transformations, leading to stable and quick generation, while transformer models effectively capture intricate long-range dependencies, making flow-based transformers suitable for this task. As shown in Figure 2 (b), based on Flag-Dit (Gao et al., 2024), we design the Band Transformer Blocks as the vector field estimator. We add the vocal encoder's output z_v to the noisy input x_t, leveraging the self-attention mechanism for alignment, and use RMSNorm (Zhang and Sennrich, 2019) and style adaptor with AdaLN (Peebles and Xie, 2023) to ensure training stability and style consistency. Additionally, we employ rotary positional embeddings (RoPE) (Su et al., 2024) to capture temporal relationships and the zero-initialized attention mechanism (Bachlechner et al., 2021) to effectively incorporate conditional information from text tokens z_p into the model. For more details, please refer to Appendix D.3.

Band-MOE To further enhance accompaniment quality, alignment, and control, we propose Band-MOE (Mixture of Experts) in the Band Transformer Block, selecting suitable experts for various conditions. As shown in Figure 3 (d), Band-MOE consists of three expert groups: Aligned MOE, Controlled MOE, and Acoustic MOE, each containing



(a) Flow-based Style Predictor (b) Band-MOE

Figure 3: The architecture of Flow-based Style Predictor and Band-MOE. Dashed lines represent optional processes. C and U represent concatenation and upsampling operations.

multiple experts. Aligned MOE conditions on z_v, adjusting inputs to match vocal features like loudness and frequency, selecting suitable experts like one specialized in large loudness and alto range. Controlled MOE uses aligned styles in text prompts to select experts for fine-grained style control, such as one for aggressive drums with metal guitar tones. Given the varying behavior of the transformer at different noise levels (Feng et al., 2023), we design a global router to adjust the weightings for Aligned MOE and Controlled MOE: 1) at early time steps (near 0), where the hidden representation h is highly noisy, the network prioritizes matching with vocal for coherent reconstruction; 2) at later time steps (near 1), where h has been largely reconstructed, the network focuses more on refining stylistic details, relying heavily on text prompts.

Finally, mel-spectrogram patterns exhibit variation across acoustic frequencies (Lee et al., 2022b). In music, high-frequency components often include the harmonics and overtones of instruments like strings and flutes. At the same time, low-frequency content typically encompasses basslines and kick drums providing rhythm and depth. Since the accomp encoder employs 1D convolutions, the latent should retain the frequency distribution. Thus, we design Acoustic MOE, selecting experts in different acoustic frequency dimensions for better quality. All routing strategies are based on the dense-to-sparse Gumbel-Softmax (Nie et al., 2021), allowing dynamic and efficient expert selection. For more details and algorithm, please refer to Appendix D.4.

Classifier-free Guidance To further control styles of the generated accompaniment based on input text prompts, we implement the classifier-free guidance (CFG) strategy. During AccompBand training, we randomly replace input text tokens z_p with encoded empty strings $\varnothing$ at a probability of 0.2. During inference, we modify the output vector field of the Band Transformer blocks as follows:

$$v_{cfg} = \gamma v_t(x, t|z_p; \theta) + (1 - \gamma)v_t(x, t|\varnothing; \theta), \quad (2)$$

where γ is the classifier free guidance scale trading off creativity and controllability. When $\gamma = 1$, v_{cfg} is the same as the original vector field $v_t(x, t|z_p; \theta)$. For more details, please refer to Appendix D.5.

3.4 Lyric and Melody Generation

LyricBand To enable more personalized tasks, we introduce LyricBand, a system designed to generate complete lyrics based on text prompts. Users can design the theme, emotion, and other parameters to generate personalized lyrics. We leverage QLoRA (Dettmers et al., 2024) for efficient fine-tuning of a well-performing open-source bilingual language model Qwen-7B (Bai et al., 2023). With 4-bit quantization and low-rank adapters, QLoRA enables LyricBand to adapt effectively to lyric generation, enabling customization and creativity. For more details, please refer to Appendix E.1.

MelodyBand Previous singing voice and song generation models often require users to provide music scores to achieve stable melodies (Zhiqing et al., 2024), lacking customization of the melody. Inspired by symbolic music models (Dong et al., 2018), we propose MelodyBand, which generates musical notes based on text prompts, lyrics, and vocal prompts. We employ a non-autoregressive transformer model to efficiently generate notes. After encoding phonemes and timbre, MelodyBand achieves fine-grained melody control by injecting text tokens through cross-attention mechanisms. We train MelodyBand with the cross-entropy loss for note pitches and an L2 loss for note durations. For more details, please refer to Appendix E.2.

3.5 Training and Inference

The VocalBand, AccompBand, LyricBand, and MelodyBand are trained separately, and the detailed training details are provided in Appendix B.2. For inference, our model can accept various prompts for multiple tasks. Without input lyrics or music scores, LyricBand and MelodyBand generate phonemes p and notes n. For song generation or singing style transfer, VocalBand generates the target vocal y_v, and AccompBand generates the target accompaniment y_a from Gaussian noise ϵ. During music style transfer, AccompBand uses noisy prompt accompaniment $\tilde{y_a}$ instead of ϵ as input. In vocal-to-song, VocalBand is not used, whereas in accompaniment-to-song, notes n extracted from GT accompaniment $\hat{y_a}$ guide VocalBand. More inference details can be found in Appendix B.3.

4 Experiments

4.1 Experimental Setup

Dataset We train our model using a combination of bilingual web-crawled and open-source song datasets. Since there are no publicly available annotated song datasets including vocals and accompaniments, we collect 20k Chinese and English songs from well-known music websites. To expand data, we also use open-source singing datasets including GTSinger (Zhang et al., 2024b) (30 hours in Chinese and English), M4Singer (Zhang et al., 2022a) (30 hours in Chinese), and OpenSinger (Huang et al., 2021) (83 hours in Chinese). After processing and cleaning, we have about 1,000 hours of song data and 1,150 hours of vocal data. We also use a filtered subset of LP-MusicCaps-MSD (Doh et al., 2023), resulting in about 1,200 hours of accompaniment data. For zero-shot evaluation, we leave out 500 out-of-domain bilingual samples with unseen singers as the test set for each task. For more details, please refer to Appendix F.

Implementation Details Mel-spectrograms are driven from raw waveforms with a 24kHz sample rate, 1280 window size, 320 hop size, and 80 mel bins. We use 4 layers of Band Transformer Blocks. The flow-matching time step is 100 for VocalBand and 1000 for AccompBand during training, while 25 during inference with the Euler ODE solver. For more details, please refer to Appendix B.1.

Evaluation Metrics We conduct both subjective and objective evaluations on generated samples. For lyric generation, we use overall quality (OVL) and relevance to the prompt (REL) for subjective evaluation. In melody generation, multiple objective metrics are employed for controllability. We use the Krumhansl-Schmuckler algorithm to predict the potential key of the generated notes and report the average key accuracy KA. We compute the average absolute difference of average pitches (APD) and temporal duration (TD, in seconds). Then, we employ pitch and duration distribution similarity (PD and DD). Melody distance (MD) is also computed using dynamic time warping.

For vocal generation, we conduct MOS (Mean Opinion Score) as the subjective evaluation. We use MOS-Q for synthesized quality and MOS-C for controllability based on text prompts. We also use F0 Frame Error (FFE) as the objective metric. For singing style transfer, we also employ MOS-S and Cosine Similarity (Cos) to assess singer similarity in timbre and personalized styles of vocal prompts.

For song generation, raters evaluate audio samples in overall quality (OVL), relevance to the prompt (REL), and alignment with the vocal (ALI). For objective evaluation, we calculate Frechet Audio Distance (FAD), Kullback–Leibler Divergence (KLD), and the CLAP score (CLAP). Please refer to Appendix G for more evaluation details.

Baseline Models For lyric generation, we use the original Qwen-7B (Bai et al., 2023) as the baseline model. For melody generation, we compare with SongMASS (Sheng et al., 2021) and MIDI part of MelodyLM (Li et al., 2024a). For vocal generation, we compare with VISinger2 (Zhang et al., 2022b), a high-fidelity SVS model, StyleSinger (Zhang et al., 2024a), a zero-shot SVS model, and vocal part of Melodist (Zhiqing et al., 2024) and MelodyLM. For song generation, we compare with Melodist and MelodyLM. For Melodist and MelodyLM, we use their papers and demos for evaluation, and open-source codes for other models. Please refer to Appendix H for more details.

4.2 Lyric and Melody Generation

Lyric Generation We evaluate lyric generation models with different text prompts covering aspects such as theme, emotion, genre, style, and specific keywords to generate lyrics. As shown in Table 3, our fine-tuned LyricBand model outperforms the original Qwen-7B model in overall quality and relevance to text prompts. This result highlights the effectiveness of our LyricBand in handling the specific downstream task more proficiently.

Methods	KA(%)↑	APD↓	TD↓	PD(%)↑	DD(%)↑	MD ↓
SongMASS	58.9	3.78	2.93	55.4	68.1	3.41
MelodyLM	**76.6**	2.05	2.29	62.8	40.8	3.62
MelodyBand	72.7	**1.74**	**1.65**	**65.8**	**70.5**	**3.12**

Table 1: Results of melody generation.

Methods	Vocal Generation			Singing Style Transfer			
	MOS-Q ↑	MOS-C ↑	FFE ↓	MOS-Q ↑	MOS-C ↑	MOS-S ↑	Cos ↑
GT	4.34 ± 0.09	-	-	4.35 ± 0.06	-	-	-
Melodist	3.83±0.09	-	0.12	-	-	-	-
MelodyLM	3.88±0.10	-	0.08	3.76±0.12	-	3.81±0.12	-
VISinger2	3.62±0.07	3.63±0.09	0.16	3.55±0.11	3.57±0.05	3.70±0.08	0.82
StyleSinger	3.90±0.08	3.96±0.05	0.08	3.87±0.06	3.86±0.09	4.05±0.05	0.89
VocalBand	**4.04±0.08**	**4.02±0.07**	**0.07**	**3.96±0.10**	**3.95±0.06**	**4.12±0.04**	**0.90**

Table 2: Results of vocal generation and singing style transfer.

Methods	OVL↑	REL↑
GT	92.31±1.29	84.07±1.63
Qwen-7B	74.35±1.37	80.66±0.92
LyricBand	**79.68±1.05**	**82.01±1.13**

Table 3: Results of lyric generation.

Melody Generation For MelodyLM, since the melody part is closed-sourced, we directly use the objective metrics reported in the paper. As shown in Table 1, MelodyBand outperforms SongMASS across all metrics and performs better than MelodyLM except KA. Since we use a non-autoregressive transformer architecture, the generation speed is much faster than the autoregressive model of MelodyLM. Thus, although MelodyLM has a slightly higher KA, our model is more suitable for the multi-task song generation system.

4.3 Vocal Generation

We evaluate VocalBand on both zero-shot vocal generation and singing style transfer tasks using the same test set with unseen singers for fair comparison. To enable style control (e.g., singing method, emotion, techniques), we incorporate our text encoder and style alignment models into VISinger2 and StyleSinger. Notably, Melodist uses known singer IDs, making it unfair for zero-shot comparisons and incapable of achieving style transfer. Meanwhile, neither Melodist nor MelodyLM control singing styles, so MOS-C is not provided.

As shown in Table 2, VocalBand consistently outperforms baseline models in both tasks, achieving higher quality (MOS-Q, FFE), similarity (MOS-S, Cos), and controllability (MOS-C). This shows the effectiveness of our Flow-based Style Predictor for style control and transfer, as well as the high quality provided by the Flow-based Pitch Predictor and Mel Decoder. For more detailed and visualized results, please refer to Appendices I.1 and I.2.

4.4 Song Generation

For song generation evaluation, we remix the generated vocals by VocalBand and accompaniments by AccompBand. For MelodyLM and Melodist, we use the objective metrics in their papers and subjectively evaluate the demos on their demo pages. We test the multi-task capabilities of AccompBand under different conditions: using LyricBand when lyrics are not provided, adding MelodyBand when music scores are missing, using both with no prompts, and finally evaluating full text prompts and optional timbre prompts are provided. Notably, the REL of MelodyLM and Melodist only considers accompaniment controllability. In contrast, for our model, we evaluate lyrics, melody, singing styles, and music styles based on text prompts.

The results are listed in Table 4, where VersBand demonstrates the highest perceptual quality (FAD, KLD, OVL), the best adherence to text prompts (CLAP, REL), and the most effective alignment (ALI). This demonstrates the quality and controllability of VocalBand, as well as the quality, controllability, and alignment of AccompBand. When some elements in text prompts are removed, VersBand can strike an impressive balance between creativity and stability. For experiments about more song generation tasks, please refer to Appendix I.

Methods	FAD ↓	KLD ↓	CLAP ↑	OVL ↑	REL ↑	ALI ↑
Melodist	3.81	1.34	0.39	84.12±1.54	85.97±1.51	74.86±1.13
MelodyLM	3.42	1.35	0.35	85.23±1.62	86.44±0.90	75.41±1.34
VersBand (w/o lyrics)	3.37	1.30	0.50	86.65±0.91	85.98±1.08	77.02±1.33
VersBand (w/o scores)	3.38	1.31	0.48	85.12±0.77	85.15±1.22	75.91±1.62
VersBand (w/o prompts)	3.55	1.35	-	83.49±1.20	-	74.87±1.68
VersBand (w/ full)	**3.01**	**1.27**	**0.58**	**87.92±1.73**	**88.03±0.59**	**80.51±1.66**

Table 4: Results of song generation. Content includes lyrics and music scores.

Methods	FAD ↓	KLD ↓	CLAP ↑	OVL ↑	REL ↑	ALI ↑
VersBand	3.01	1.27	0.58	87.92±1.73	88.03±0.59	80.51±1.66
w/o Band-MOE	3.29	1.34	0.42	86.11±1.30	87.49±0.84	77.59±1.50
w/o Aligned MOE	3.15	1.25	0.54	87.19±1.14	88.48±0.66	77.86±1.35
w/o Controlled MOE	3.10	1.23	0.44	88.48±1.74	87.90±1.50	79.33±1.63
w/o Acoustic MOE	3.26	1.32	0.40	86.50±1.55	87.72±1.04	79.08±1.24

Table 5: Results of ablation study on AccompBand.

Methods	MOS-Q↑	MOS-C↑	FEE↓
VocalBand	4.04±0.08	4.02±0.07	0.07
w/o styles	3.87±0.04	-	0.09
w/o Pirch Predictor	3.79±0.06	3.99±0.09	0.09
w/o Mel Decoder	3.68±0.08	3.92±0.07	0.13

Table 6: Ablation Results of VocalBand.

4.5 Ablation Study

Ablation Study on VocalBand We conduct tests on key modules of VocalBand. To compare quality, we remove the style information from the Flow-based Style Predictor, and replace the Flow-based Pitch Predictor and Mel Decoder with simpler models from FastSpeech2 (Ren et al., 2020) for comparison. As shown in Table 6, we observe that the absence of style representation leads to a decrease in quality, as it cannot generate vocals with rich emotional and stylistic variations, nor can it achieve style control or style transfer. Additionally, our Flow-based Pitch Predictor and Mel Decoder contribute significantly to the overall quality.

Ablation Study on AccompBand We conduct tests on major modules of AccompBand. We set the full Band-MOE and three expert groups removed as other baseline models. As shown in Table 5, removing the Band-MOE results in a decline in all metrics. For individual expert groups, we observe that the Aligned MOE affects alignment, while the Controlled MOE impacts controllability. The absence of the Acoustic MOE, which handles different acoustic channels, leads to a drop in quality.

Ablation Study on VersBand We remove various components from text prompts for evaluation. As shown in Table 4, even with a minimum input, VersBand still delivers remarkable performance. When listening to songs generated for various tasks on our demo page, it is evident VersBand shows strong controllability and expressiveness over various text prompts, along with the ability to produce intricate, skillful vocals employing multiple techniques, and complex, well-aligned accompaniments featuring harmonious instrumentation. For more ablation studies, please refer to Appendix J.

5 Conclusions

In this paper, we present VersBand, a multi-task song generation framework for synthesizing high-quality, aligned songs with prompt-based control. We mainly design these models: 1) VocalBand, a decoupled model leveraging the flow-matching model for singing styles, pitches, and mel-spectrograms generation, allowing fast and high-quality vocal generation with style control. 2) AccompBand, a flow-based transformer model, incorporates the Band-MOE, selecting suitable experts for enhanced quality, alignment, and control. This model generates controllable, high-quality accompaniments aligned with vocals. 3) Two generation models, LyricBand for lyrics and MelodyBand for melodies, contribute to the comprehensive multi-task song generation system. Experimental results demonstrate that VersBand performs better over baseline models across multiple song generation tasks using objective and subjective metrics.

6 Limitations

In this section, we discuss two main limitations of VersBand and provide potential strategies to address them in future work: 1) To achieve comprehensive controllability and high-quality multi-task song generation based on various prompts, VersBand utilizes four sub-models to generate different song components, relying on multiple infrastructures like the flow-based transformer and VAE. This results in cumbersome training and inference procedures. Future work will explore using a single model to achieve the same multi-task generation capabilities and controllability. 2) Our dataset only includes songs in Chinese and English, lacking diversity. In the future, we will attempt to build a larger and more comprehensive dataset to enable a wider range of application scenarios.

7 Ethics Statement

Large-scale generative models always present ethical challenges. VersBand, due to its multi-task song generation capabilities, could potentially be misused for dubbing in entertainment short videos, raising concerns about the infringement of famous singers' copyrights. Then, its ability to transfer and control multiple song styles about lyric, melody, singing, and music, lowers the requirements for high-quality, personalized, controllable song generation, posing some risks like unfair competition and potential unemployment for professionals in related music and singing occupations. To mitigate these potential risks, we will explore methods like music watermarking to protect individual privacy.

References

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

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

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. 2021. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in Artificial Intelligence*, pages 1352–1361. PMLR.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

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

Max Bain, Jaesung Huh, Tengda Han, and Andrew Zisserman. 2023. Whisperx: Time-accurate speech transcription of long-form audio. *INTERSPEECH 2023*.

Donald J Berndt and James Clifford. 1994. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:.

Jiawei Chen, Xu Tan, Jian Luan, Tao Qin, and Tie-Yan Liu. 2020. Hifisinger: Towards high-fidelity neural singing voice synthesis. *arXiv preprint arXiv:2009.01776*.

Ke Chen, Yusong Wu, Haohe Liu, Marianna Nezhurina, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2024. 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)*, pages 1206–1210. IEEE.

Yin-Ping Cho, Yu Tsao, Hsin-Min Wang, and Yi-Wen Liu. 2022. Mandarin singing voice synthesis with denoising diffusion probabilistic wasserstein gan. *Preprint*, arXiv:2209.10446.

Soonbeom Choi and Juhan Nam. 2022. A melody-unsupervision model for singing voice synthesis. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7242–7246. IEEE.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2024. Simple and controllable music generation. *Advances in Neural Information Processing Systems*, 36.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Shuangrui Ding, Zihan Liu, Xiaoyi Dong, Pan Zhang, Rui Qian, Conghui He, Dahua Lin, and Jiaqi Wang. 2024. Songcomposer: A large language model for lyric and melody composition in song generation. *arXiv preprint arXiv:2402.17645*.

SeungHeon Doh, Keunwoo Choi, Jongpil Lee, and Juhan Nam. 2023. Lp-musiccaps: Llm-based pseudo music captioning. *arXiv preprint arXiv:2307.16372*.

Chris Donahue, Antoine Caillon, Adam Roberts, Ethan Manilow, Philippe Esling, Andrea Agostinelli, Mauro Verzetti, Ian Simon, Olivier Pietquin, Neil Zeghidour, et al. 2023. Singsong: Generating musical accompaniments from singing. *arXiv preprint arXiv:2301.12662*.

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

Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. 2023. 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.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Zhida Feng, Zhenyu Zhang, Xintong Yu, Yewei Fang, Lanxin Li, Xuyi Chen, Yuxiang Lu, Jiaxiang Liu, Weichong Yin, Shikun Feng, et al. 2023. Ernie-vilg 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10135–10145.

Peng Gao, Le Zhuo, Ziyi Lin, Chris Liu, Junsong Chen, Ruoyi Du, Enze Xie, Xu Luo, Longtian Qiu, Yuhang Zhang, et al. 2024. Lumina-t2x: Transforming text into any modality, resolution, and duration via flow-based large diffusion transformers. *arXiv preprint arXiv:2405.05945*.

Qingqing Huang, Aren Jansen, Joonseok Lee, Ravi Ganti, Judith Yue Li, and Daniel PW Ellis. 2022. Mulan: A joint embedding of music audio and natural language. *arXiv preprint arXiv:2208.12415*.

Rongjie Huang, Feiyang Chen, Yi Ren, Jinglin Liu, Chenye Cui, and Zhou Zhao. 2021. Multi-singer: Fast multi-singer singing voice vocoder with a large-scale corpus. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3945–3954.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

Ziyue Jiang, Jinglin Liu, Yi Ren, Jinzheng He, Zhenhui Ye, Shengpeng Ji, Qian Yang, Chen Zhang, Pengfei Wei, Chunfeng Wang, et al. 2024. Mega-tts 2: Boosting prompting mechanisms for zero-shot speech synthesis. In *The Twelfth International Conference on Learning Representations*.

Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. 2018. Fr\'echet audio distance: A metric for evaluating music enhancement algorithms. *arXiv preprint arXiv:1812.08466*.

Tae-Woo Kim, Min-Su Kang, and Gyeong-Hoon Lee. 2024. Adversarial multi-task learning for disentangling timbre and pitch in singing voice synthesis. *Preprint*, arXiv:2206.11558.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in neural information processing systems*, 33:17022–17033.

Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. 2022a. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11523–11532.

Sang-gil Lee, Wei Ping, Boris Ginsburg, Bryan Catanzaro, and Sungroh Yoon. 2022b. Bigvgan: A universal neural vocoder with large-scale training. *arXiv preprint arXiv:2206.04658*.

Shun Lei, Yixuan Zhou, Boshi Tang, Max WY Lam, Feng Liu, Hangyu Liu, Jingcheng Wu, Shiyin Kang, Zhiyong Wu, and Helen Meng. 2024. Songcreator: Lyrics-based universal song generation. *arXiv preprint arXiv:2409.06029*.

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

Ruiqi Li, Yu Zhang, Yongqi Wang, Zhiqing Hong, Rongjie Huang, and Zhou Zhao. 2024b. Robust singing voice transcription serves synthesis. *Preprint*, arXiv:2405.09940.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. 2022. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*.

Xingchao Liu, Chengyue Gong, et al. 2022. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*.

Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, volume 2017, pages 498–502.

Xiaonan Nie, Xupeng Miao, Shijie Cao, Lingxiao Ma, Qibin Liu, Jilong Xue, Youshan Miao, Yi Liu, Zhi Yang, and Bin Cui. 2021. Evomoe: An evolutional mixture-of-experts training framework via dense-to-sparse gate. *arXiv preprint arXiv:2112.14397*.

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205.

Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark Hasegawa-Johnson. 2019. Autovc: Zero-shot voice style transfer with only autoencoder loss. In *International Conference on Machine Learning*, pages 5210–5219. PMLR.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2020. Fastspeech 2: Fast and high-quality end-to-end text to speech. *arXiv preprint arXiv:2006.04558*.

BS Series. 2011. Algorithms to measure audio programme loudness and true-peak audio level. In *International Telecommunication Union Radiocommunication Assembly*.

Zhonghao Sheng, Kaitao Song, Xu Tan, Yi Ren, Wei Ye, Shikun Zhang, and Tao Qin. 2021. Songmass: Automatic song writing with pre-training and alignment constraint. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13798–13805.

Jiatong Shi, Shuai Guo, Tao Qian, Nan Huo, Tomoki Hayashi, Yuning Wu, Frank Xu, Xuankai Chang, Huazhe Li, Peter Wu, Shinji Watanabe, and Qin Jin. 2022. Muskits: an end-to-end music processing toolkit for singing voice synthesis. *Preprint*, arXiv:2205.04029.

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

Silero Team. 2021. Silero vad: pre-trained enterprise-grade voice activity detector (vad), number detector and language classifier. *Retrieved March*, 31:2023.

Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu, et al. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 12.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yongqi Wang, Ruofan Hu, Rongjie Huang, Zhiqing Hong, Ruiqi Li, Wenrui Liu, Fuming You, Tao Jin, and Zhou Zhao. 2024. Prompt-singer: Controllable singing-voice-synthesis with natural language prompt. *arXiv preprint arXiv:2403.11780*.

Haojie Wei, Xueke Cao, Tangpeng Dan, and Yueguo Chen. 2023. Rmvpe: A robust model for vocal pitch estimation in polyphonic music. *arXiv preprint arXiv:2306.15412*.

Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2023. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. 2021. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*.

Biao Zhang and Rico Sennrich. 2019. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32.

Lichao Zhang, Ruiqi Li, Shoutong Wang, Liqun Deng, Jinglin Liu, Yi Ren, Jinzheng He, Rongjie Huang, Jieming Zhu, Xiao Chen, et al. 2022a. M4singer: A multi-style, multi-singer and musical score provided mandarin singing corpus. *Advances in Neural Information Processing Systems*, 35:6914–6926.

Yongmao Zhang, Heyang Xue, Hanzhao Li, Lei Xie, Tingwei Guo, Ruixiong Zhang, and Caixia Gong. 2022b. Visinger 2: High-fidelity end-to-end singing voice synthesis enhanced by digital signal processing synthesizer. *Preprint*, arXiv:2211.02903.

Yu Zhang, Rongjie Huang, Ruiqi Li, JinZheng He, Yan Xia, Feiyang Chen, Xinyu Duan, Baoxing Huai, and Zhou Zhao. 2024a. Stylesinger: Style transfer for out-of-domain singing voice synthesis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19597–19605.

Yu Zhang, Changhao Pan, Wenxiang Guo, Ruiqi Li, Zhiyuan Zhu, Jialei Wang, Wenhao Xu, Jingyu Lu, Zhiqing Hong, Chuxin Wang, et al. 2024b. Gtsinger: A global multi-technique singing corpus with realistic music scores for all singing tasks. *arXiv preprint arXiv:2409.13832*.

Hong Zhiqing, Huang Rongjie, Cheng Xize, Wang Yongqi, Li Ruiqi, You Fuming, Zhao Zhou, and Zhang Zhimeng. 2024. Text-to-song: Towards controllable music generation incorporating vocals and accompaniment. *arXiv preprint arXiv:2404.09313*.

A Rectified flow-matching

In this section, we introduce the flow-matching generative method, as described by Liu et al. (2022). In generative modeling, the true data distribution is denoted as $q(x_1)$, which can be sampled but lacks an accessible density function. Consider a probability path $p_t(x_t)$, where $x_0 \sim p_0(x)$ represents a known simple distribution (e.g., a standard Gaussian), and $x_1 \sim p_1(x)$ approximates the real data distribution. The objective of flow-matching is to model this probability path directly, expressed as an ordinary differential equation (ODE):

$$\mathrm{d}x = u(x,t)\mathrm{d}t, \quad t \in [0,1], \tag{3}$$

where $u(x,t)$ denotes the target vector field, and t is the time index. If the vector field u is known, realistic data can be recovered by reversing the flow. To approximate u, a vector field estimator $v(\cdot)$ is used, with the flow-matching objective defined as:

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t,p_t(x)} \left\| v(x,t;\theta) - u(x,t) \right\|^2, \tag{4}$$

where $p_t(x)$ denotes the distribution of x at time t. To enable conditional generation, we add conditional information c, leading to the conditional flow-matching objective (Lipman et al., 2022):

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,p_1(x_1),p_t(x|x_1)} \left\| v(x,t|c;\theta) - u(x,t|x_1,c) \right\|^2. \tag{5}$$

Flow-matching proposes a straight path from noise to data. Specifically, we use a linear interpolation between the data x_1 and Gaussian noise x_0 to generate samples at time t:

$$x_t = (1-t)x_0 + tx_1. \tag{6}$$

Thus, the conditional vector field becomes $u(x,t|x_1,c) = x_1 - x_0$, and the rectified flow-matching (RFM) loss used for gradient descent is:

$$\left\| v(x,t|c;\theta) - (x_1 - x_0) \right\|^2. \tag{7}$$

If the vector field u is estimated correctly, we can generate realistic data by propagating Gaussian noise through an ODE solver at discrete time steps. A widely used method for solving the reverse flow is the Euler ODE:

$$x_{t+\epsilon} = x + \epsilon v(x,t|c;\theta), \tag{8}$$

where ϵ is the step size. In our VocalBand, we use content, timbre, prompt style, text tokens, and other inputs for each task as conditioning information c, while the target data x_1 consists of target style, F0, or mel-spectrograms. In our AccompBand, we use timestep, text tokens, and vocal embedding as conditioning information c, while the target data x_1 is the accompaniment embedding.

Moreover, flow matching models require 100 to 1000 steps during training, but since they generate a straight path, they only require 25 or fewer steps during inference, making the generation highly efficient for fast generation. Additionally, flow-matching models ensure stable and high-quality generation due to their ability to model smooth transitions between noise and data, maintaining fidelity throughout the process. This stability is crucial for complex generation tasks, as it reduces artifacts and enhances the consistency of the output across various conditions.

B VersBand Details

B.1 Model Details

Our VersBand framework consists of four models: VocalBand, AccompBand, LyricBand, and MelodyBand. For the text encoder, we use FLAN-T5-large (Chung et al., 2024), while we also test BERT-large (Devlin et al., 2018) and the text encoder of CLAP (Elizalde et al., 2023) in Appendix J.1. Our vocoder is the pre-trained HiFi-GAN (Kong et al., 2020). For detailed hyperparameters of each component, please refer to Appendix C.1, D.1, E.1, and E.2.

For training details, we set the sample rate to 24kHz, the window size to 1280, the hop size to 320, and the number of mel bins to 80 to derive mel-spectrograms from raw waveforms. We train VocalBand on 4 NVIDIA RTX-4090 GPUs for 200k steps. The Adam optimizer is used with $\beta_1 = 0.9$ and $\beta_2 = 0.98$. AccompBand is trained on 8 NVIDIA RTX-4090 GPUs for 80k steps, using the AdamW optimizer with a base learning rate of 3×10^{-6}. The pre-trained accomp encoder and decoder are trained on 4 NVIDIA RTX-4090 GPUs for 40k steps. MelodyBand is trained for 30k steps until convergence on 4 NVIDIA RTX-4090 GPUs. LyricBand is fine-tuned for 4k steps until convergence on 4 NVIDIA RTX-4090 GPUs.

B.2 Training Procedures

For VocalBand, the final loss terms in the training phase include the following components: 1) $\mathcal{L}_{commit}$: the commitment loss for the residual style encoder in Equation 9; 2) $\mathcal{L}_{style}$: the flow

matching loss of Flow-based Style Predictor in Equation 1; 3) $\mathcal{L}_{pitch}$: the flow matching loss of Flow-based Pitch Predictor; 4) $\mathcal{L}_{mel}$: the flow matching loss of Flow-based Mel Decoder; 5) $\mathcal{L}_{dur}$: the MSE duration loss between the predicted and the GT phoneme-level duration in the log scale.

As for AccompBand, the final loss terms during training consist of the following aspects: 1) $\mathcal{L}_{balance}$: the load-balancing loss for each expert group in Band-MOE in Equation 14; 2) $\mathcal{L}_{flow}$: the flow matching loss of AccompBand.

For the pre-trained accomp encoder and accomp decoder, the final loss terms include: 1) $\mathcal{L}_{rec}$: the L2 reconstruction loss; 2) $\mathcal{L}_{adv}$: the LSGAN-styled adversarial loss in GAN discriminator.

Regarding MelodyBand, the final loss terms for training involve: 1) $\mathcal{L}_{pitch}$: the cross-entropy loss for note pitches in Equation 15; 2) $\mathcal{L}_{duration}$: the L2 loss for note durations in Equation 16.

B.3 Multi-Task Inference Procedures

During inference, we can achieve multiple song generation tasks based on text and audio prompts. If full lyrics are not provided, LyricBand generates phonemes p based on the text tokens z_p. Without input music scores, MelodyBand generates notes n (note pitches and note durations) based on lyrics, text prompts, and optional vocal prompts.

For the song generation task, VocalBand generates the target vocal y_v based on n and p as contents, along with z_p to control styles, timbre prompt z_t is optional. AccompBand generates the target accompaniment y_a from Gaussian noise ϵ and y_v.

To conduct singing style transfer, VocalBand additionally takes a vocal prompt $\tilde{y_a}$ as input to extract timbre z_t and prompt style $\tilde{z_s}$. The target vocal is required to maintain consistent timbre and personal style (e.g., pronunciation, articulation skills). The Flow-based Style Predictor is used to predict the target style z_s, learning both personalized styles from $\tilde{z_s}$ and style control information from z_p (such as singing methods). Notably, $\tilde{z_s}$ and z_p can be input individually for full style control.

For music style transfer, AccompBand uses the noisy prompt accompaniment $\tilde{y_a}$ with a time step 0.5 instead of ϵ and sums it with target vocal y_v, enabling the model to learn the style from the retained components of the prompt accompaniment.

In the vocal-to-song task, the GT vocal is used to guide AccompBand in generating the accompaniment. In contrast, for the accompaniment-to-song task, notes n are extracted from the GT accompaniment $\hat{y_a}$ using ROSVOT (Li et al., 2024b) to guide VocalBand in vocal generation, while AccompBand is not used.

Hyper-parameter		Value
Phoneme Encoder	Phoneme Embedding	256
	Encoder Layers	4
	Encoder Hidden	256
	Encoder Conv1D Kernel	9
	Encoder Conv1D Filter Size	1024
Note Encoder	Pitch Embedding	256
	Duration Hidden	256
Timbre Encoder	Encoder Layers	5
	Hidden Size	256
	Conv1D Kernel	31
Residual Style Encoder	Conv Layers	5
	RQ Codebook Size	256
	Depth of RQ	4
Flow-based Style Predictor	Conv Layers	20
	Kernel Size	3
	Residual Channel	256
	Hidden Channel	256
	Training Time Steps	100
Flow-based Pitch Predictor	Conv Layers	12
	Kernel Size	3
	Residual Channel	192
	Hidden Channel	256
	Training Time Steps	100
Flow-based Mel Decoder	Conv Layers	20
	Kernel Size	3
	Residual Channel	256
	Hidden Channel	256
	Training Time Steps	100
Total Number of Parameters		56.26M

Table 7: Hyper-parameters of VocalBand.

C VocalBand Details

C.1 Model Configuration

We list the architecture and hyperparameters of VocalBand in Table 7.

C.2 Decomposition Strategy

We assume that the target vocal y_v can be decomposed into three distinct representations: content z_c, style z_s (e.g., singing methods, emotion, techniques, pronunciation, and articulation skills), and timbre z_t. When a vocal prompt $\tilde{y_v}$ is provided during training, our goal is to transfer both the timbre $\tilde{z_t}$ and personalized style $\tilde{z_s}$ (like pronunciation and articulation skills) from the vocal prompt to the target vocal y_v. Meanwhile, we also need to achieve style control from text tokens z_p (such as singing method, emotion, and techniques).

Following previous style transfer approaches

(Jiang et al., 2024), we assume that the mutual information between y_v and $\tilde{y_v}$ primarily captures global information, represented by z_t (timbre). Therefore, the target timbre z_t is set equal to the prompt timbre $\tilde{z_t}$, as we aim to control the timbre of the output based on the user's input. Under this assumption, $\tilde{z_t}$ is extracted using a timbre encoder, which focuses solely on timbre information, without capturing style z_s or content z_c. To ensure that the content encoders extract only content-related information, we feed it phoneme sequences and musical notes, allowing it to exclusively pass the content representation z_c. For the timbre and content encoders, please refer to Appendix C.3 and C.4.

Once both z_c and z_t are obtained, we must remove fine-grained content and timbre information from the target style z_s. We employ a residual style encoder to extract the prompt style $\tilde{z_s}$, and then use the Flow-based Style Predictor to predict the target style z_s. The latent vector z_s generated by the Flow-based Style Predictor not only captures the personalized styles consistent with the prompt style $\tilde{z_s}$ (e.g., pronunciation and articulation skills) but also incorporates the styles in the text tokens z_p (like singing methods, emotions, and techniques).

By utilizing a residual quantization (RQ) model (Lee et al., 2022a) as an information bottleneck (Qian et al., 2019), the residual style encoder is compelled to transmit only the fine-grained style information z_s (Zhang et al., 2024a), which other encoders cannot capture. Both z_s and $\tilde{z_s}$ share the same form as the RQ embeddings, consisting of multiple layers of fine-grained style information that are disentangled from both timbre and content. This is because z_s is the output of the flow-matching ODE solver, whose training objective is to capture the target style from the ground truth vocals, as extracted by the residual style encoder. For more details about the Flow-based Style Predictor, please refer to Appendix C.6. Consequently, the process guarantees the successful decomposition of style from content and timbre. These embeddings z_c, z_t, and z_s are then fed into a duration predictor (Ren et al., 2020) and a length regulator for subsequent F0 and mel-spectrogram prediction.

C.3 Timbre Encoder

The timbre encoder, designed to capture the unique identity of the singer, extracts a global timbre vector $\tilde{z_t}$ from the vocal prompt $\tilde{y_v}$. The encoder consists of multiple stacked convolutional layers. To ensure stability in the timbre representation, the output of the timbre encoder is temporally averaged, producing a one-dimensional timbre vector $\tilde{z_t}$. The target timbre z_t is set equal to the prompt timbre $\tilde{z_t}$, as we aim to control the timbre of the output based on the user's input.

C.4 Content Encoders

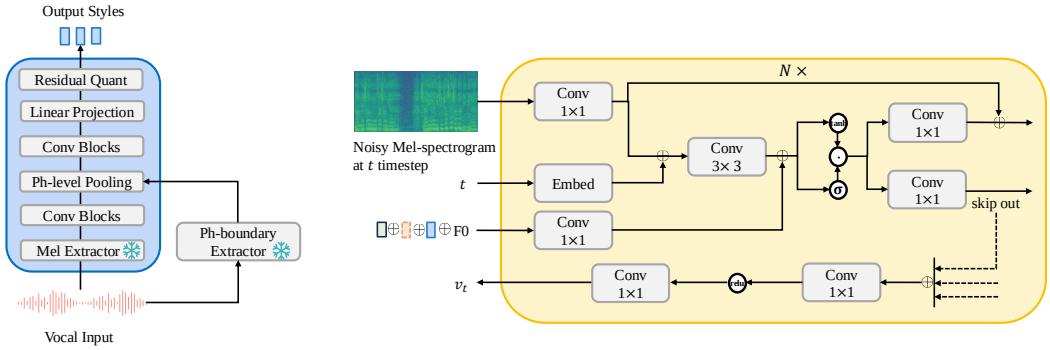
Our content encoders consist of a phoneme encoder and a note encoder. The phoneme encoder processes a sequence of phonemes p through a phoneme embedding layer followed by four FFT blocks, extracting phoneme features. In parallel, the note encoder handles musical score information n, processing note pitches and durations. These are passed through two separate embedding layers and a linear projection layer, which generate the corresponding note features. The outputs of the phoneme encoder and the note encoder are then summed as z_c.

C.5 Residual Style Encoder

Singing style can vary across and within phonemes. To comprehensively capture phoneme-level styles (such as singing methods, emotion, techniques, pronunciation, and articulation skills) and disentangle them from timbre and content, we design the residual style encoder. In the residual style encoder, we employ a Residual Quantization (RQ) module (Lee et al., 2022a) to extract singing style, creating an information bottleneck that effectively filters out non-style information (Zhang et al., 2024a). Thanks to the RQ's ability to extract multiple layers of information, it enables more comprehensive modeling of style across various hierarchical levels. Specifically, pronunciation and articulation skills encompass pitch transitions between musical notes and vibrato within a phoneme, where the multi-level modeling capability of RQ is highly suitable.

More concretely, as illustrated in Figure 4 (a), we first extract the mel-spectrogram from the input vocal using the open-source tool librosa [1] and further refine it through convolutional blocks. The output is then condensed into phoneme-level hidden states via a pooling layer, which operates based on phoneme boundaries. We utilize open-source tools including WhisperX (Bain et al., 2023) and Montreal Forced Aligner (MFA) (McAuliffe et al., 2017) to extract these phoneme boundaries directly from the input vocal. Subsequently, the convolution stacks capture phoneme-level correlations. Next,

[1] https://github.com/librosa/librosa



(a) Residual Style Encoder

(b) Vector Field Estimator

Figure 4: The architecture of two components of VocalBand, Figure (a) shows the residual style encoder while Figure (b) illustrates the vector field estimator of the Flow-based Mel Decoder.

we use a linear projection to map the output into a low-dimensional latent variable space for code index lookup, significantly enhancing the utilization of the codebook (Yu et al., 2021).

With a quantization depth of n, the RQ module represents the input z_e as a sequence of N ordered codes. Let $RQ_i(z_e)$ denote the process of representing z_e as RQ code and extracting the code embedding in the i-th codebook. The representation of z_e in the RQ module at depth $n \in [N]$ is denoted as $\hat{z_e}^n = \sum_{i=1}^{n} RQ_i(z_e)$. To ensure that the input representation adheres to a discrete embedding, a commitment loss (Lee et al., 2022a) is employed:

$$\mathcal{L}_{commit} = \sum_{n=1}^{N} \|z_e - sg[\hat{z_e}^n]\|_2^2, \tag{9}$$

where the notation sg represents the stop-gradient operator. It is important to note that $\mathcal{L}_{commit}$ is the cumulative sum of quantization errors across all n iterations, rather than a single term. The objective is to ensure that $\hat{z_e}^n$ progressively reduces the quantization error of z_e as the value of n increases. Finally, we extract the phoneme-level style embedding from the input vocal.

C.6 Flow-based Style Predictor

As shown in Figure 3 (a), the Flow-based Style Predictor uses content z_c, timbre z_t, phoneme-level prompt style $\tilde{z_s}$, and text tokens z_p to predict the target style z_s. With the combined z_c and z_t, we employ a style alignment model utilizing the Scaled Dot-Product Attention mechanism (Vaswani et al., 2017) to align style control information from z_p (e.g., singing methods, emotions, techniques) with the content. Positional embedding is applied before feeding z_p into the attention module. In the attention module, the combined z_c and z_t serve as the query z_{ct}, while z_p serves as both the key and value, and d represents the dimensionality of the key and query:

$$\begin{aligned} &Attention(Q, K, V) = Attention(z_{ct}, z_p, z_p) \\ &= Softmax\left(\frac{z_{ct} z_p^T}{\sqrt{d}}\right) z_p. \end{aligned} \tag{10}$$

We stack the style alignment layer multiple times for better performance and gradually stylize the query value. We combine the output with z_{ct} as condition c and then feed it into an ODE solver, which transforms Gaussian noise ϵ into z_s along a probability path $p_t(z_{st})$. We concatenate $\tilde{z_s}$ with ϵ to allow z_s to learn personalized styles (e.g., pronunciation and articulation skills).

During training, we set $u(z_{st}, t)$ to represent the target vector field at time t, obtained through linear interpolation between ϵ and the ground truth (GT) phoneme-level style z_s, which is extracted from the GT vocal by the residual style encoder. To stabilize the flow-matching training process, we do not train the Flow-based Style Predictor during the early stages of training (the first 50,000 steps). Instead, we feed the GT style z_s into the subsequent Flow-based Pitch Predictor and Mel Decoder. Therefore, by the time we begin training the Flow-based Style Predictor, the residual style encoder has stabilized, ensuring a consistent GT z_s, which is beneficial for the flow-matching training.

The learned vector field $v(z_{st}, t|c; \theta)$, predicted by a vector field estimator at each time t, ensures smooth interpolation between the initial noise and the output z_s, guided by the flow-matching objective. We use the non-causal WaveNet architecture

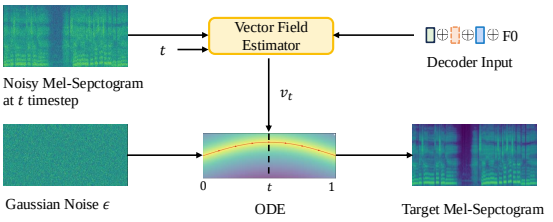


Figure 5: The architecture of Flow-based Mel Decoder.

(Van Den Oord et al., 2016) as the backbone of our vector field estimator, due to its proven capability in modeling sequential data. For more details about the vector field estimator, please refer to Appendix C.7. Notably, to enable the model to handle cases without a vocal prompt, we drop vocal prompts with a probability of 0.2 during training. We also replace z_p with embedded empty strings in a probability of 0.1 for cases without prompts.

During inference, the ODE solver generates the phoneme-level target style z_s directly from the concatenation of Gaussian noise and $\tilde{z_s}$ (if a vocal prompt is provided), based on the condition c. This method ensures fast and controllable generation of z_s, learning personalized styles consistent with $\tilde{z_s}$ while incorporating the aligned style control information from z_p.

C.7 Vector Field Estimator

We adopt the non-causal WaveNet architecture (Van Den Oord et al., 2016) as the backbone of our vector field estimators for the Flow-based Style Predictor, Pitch Predictor, and Mel Decoder, due to its demonstrated effectiveness in modeling sequential data. The architecture of the vector field estimator for the Flow-based Mel Decoder is depicted in Figure 4 (b). We input content z_c, timbre z_t, style z_s, and F0 as conditioning factors to predict the corresponding vector field. Similarly, the architecture of the vector field estimators for the Flow-based Pitch Predictor and Style Predictor follows the same structure, while the only difference lies in the input and condition for each model.

C.8 Flow-based Pitch Predictor and Mel Decoder

During training, our target F0 is extracted using the open-source tool RMVPE (Wei et al., 2023), while mel-spectrograms are extracted using the open-source tool librosa [1]. As shown in Figure 5, the Flow-based Mel Decoder employs a flow-matching architecture (Liu et al., 2022), where the vector field estimator and ODE solver generate the target mel-spectrogram from Gaussian noise ϵ. The Flow-based Pitch Predictor follows a similar flow-matching procedure. We adopt the non-causal WaveNet architecture (Van Den Oord et al., 2016) as the backbone of our vector field estimator. For further details on the vector field estimator, please refer to Appendix C.7.

Hyperparameter		Value
Accomp Encoder	Encoder Layers	3
	Encoder Hidden	384
	Encoder Conv1D Kernel	5
	Encoder Output Channels	20
Accomp Decoder	Decoder Layers	3
	Decoder Hidden	384
	Decoder Conv1D Kernel	5
	Decoder Input Channels	20
Vocal Encoder	Encoder Layers	3
	Encoder Hidden	384
	Encoder Conv1D Kernel	5
	Encoder Output Channels	20
Band Transformer Blocks	Transformer Layers	4
	Transformer Embed Dim	768
	Transformer Attention Headers	8
	Experts for each group	4
	Training Time Steps	1000
Total Number of Parameters		431.07M

Table 8: Hyper-parameters of AccompBand.

D AccompBand Details

D.1 Model Configuration

We list the architecture and hyperparameters of AccompBand in Table 8.

D.2 Accomp Encoder and Decoder

The accomp encoder and decoder are based on the VAE model (Kingma and Welling, 2013). For pre-training the accomp encoder and decoder, we use the L2 reconstruction loss: $\mathcal{L}_{rec} = \|y_v - \hat{y_v}\|^2$,where y_v is the reconstructed accompaniment mel-spectrogram and $\hat{y_v}$ is the ground truth accompaniment mel-spectrogram. Additionally, we incorporate a GAN discriminator, following the architecture of ML-GAN (Chen et al., 2020), to further enhance the quality of the reconstruction. We apply the LSGAN-style adversarial loss (Mao et al., 2017), $\mathcal{L}_{adv}$, which aims to minimize the distributional distance between the predicted mel-spectrograms and the ground truth mel-spectrograms. Before feeding the waveform into the accomp encoder, we first extract the mel-spectrogram using librosa [1]. After generating the

mel-spectrogram from the decoder output, we utilize HiFi-GAN (Kong et al., 2020) to convert it back into audio.

D.3 Band Transformer Blocks

As shown in Figure 2 (c), the Band Transformer Blocks are based on Flag-Dit (Gao et al., 2024). During training, the vocal embedding z_v extracted by the vocal encoder is added to the noisy input x_t to leverage the transformer's self-attention mechanism, allowing the model to learn vocal-matching style, rhythm, and melody. We use RMSNorm (Zhang and Sennrich, 2019) to improve training stability, preventing the absolute values from growing uncontrollably and causing numerical instability. Next, we compute the global style embedding z_g by averaging the text tokens z_p and vocal embedding z_v along the temporal dimension and adding the time step embedding of t. This global style embedding is used in a multi-layer style adaptor, which modulates the latent representation using adaptive layer normalization (AdaLN) (Peebles and Xie, 2023) to ensure style consistency. We compute the scale and shift using linear regression:

$$AdaLN(h, c) = \gamma_c \times LayerNorm(h) + \beta_c, \tag{11}$$

where h represents the hidden representation. We zero-initialize the batch norm scale factor γ in each block (Peebles and Xie, 2023). Moreover, we explore relative positional encoding with rotary positional embedding (RoPE) (Su et al., 2024), which injects temporal positional information into the model. This enables the model to capture the temporal relationships between successive frames, providing significant performance improvements for the transformer.

Then, the zero-initialized attention mechanism (Bachlechner et al., 2021) is used to inject conditional information from the text tokens z_p into the hidden states h, while simultaneously learning the vocal style, rhythm, and melody aligned with the vocal embedding z_v added to x_t. Given the accompaniment queries Q_h, keys K_h, and values V_h from hidden states, along with the text keys K_z and values V_z, the final attention output is:

$$\begin{aligned} Attention = softmax\left(\frac{\tilde{Q_h}\tilde{K_h}^\top}{\sqrt{d}}\right)V_h + \\ \tanh(\alpha)softmax\left(\frac{\tilde{Q_h}K_z^\top}{\sqrt{d}}\right)V_z, \end{aligned} \tag{12}$$

where $\tilde{Q_h}$ and $\tilde{K_h}$ denote using RoPE in both queries and keys, d is the dimensionality of both queries and keys, and α is a zero-initialized learnable parameter that gates the cross-attention with the input text tokens.

D.4 Band-MOE

As illustrated in Figure 3(d), Band-MOE is composed of three expert groups: Aligned MOE, Controlled MOE, and Acoustic MOE, each comprising multiple experts. We employ Feed-Forward Networks (FFNs) as the architecture for each expert. It is well-established (Lee et al., 2022b) that mel-spectrogram details exhibit different patterns across various acoustic frequencies. In musical accompaniment, high-frequency components often include the harmonics and overtones of instruments like strings and flutes, as well as percussive elements such as cymbals and hi-hats, which enhance the brightness and clarity of the sound. Conversely, low-frequency content encompasses basslines and kick drums, providing foundational rhythm and depth that shape the overall groove and warmth of the music. Motivated by this, previous works (Kong et al., 2020) have adopted multi-scale architectures to model downsampled signals at different frequency bands, which effectively control the periodic elements of the signal and reduce artifacts.

Building on this idea, we introduce Acoustic MOE, where experts are assigned to specific acoustic frequency bands based on the processed hidden representation h, and their outputs are aggregated to produce the final result. Moreover, since the vocal and accomp encoder employ 1D convolutions to encode both the vocal and accompaniment mel-spectrograms, the latent representation of the hidden h should retain the frequency distribution.

Our routing strategy for all routers is based on the dense-to-sparse Gumbel-Softmax method (Nie et al., 2021), enabling dynamic and efficient expert selection. The Gumbel-Softmax trick facilitates sampling from a categorical distribution by reparameterizing categorical variables to make them differentiable. Specifically, the routing score $g(h)$ for each expert i is computed as follows:

$$g(h)_i = \frac{\exp((h \cdot W_g + \zeta_i)/\tau)}{\sum_{j=1}^{N} \exp((h \cdot W_g + \zeta_j)/\tau)}, \tag{13}$$

where W_g is the learned gating weight, ζ is sampled from the Gumbel(0, 1) distribution (Jang et al., 2016), and τ is the softmax temperature. Initially,

a high temperature τ results in denser expert selection, allowing multiple experts to process the same input. As training progresses, τ is gradually decreased, making the routing sparser and selecting fewer experts for each input. When $\tau \to 0$, the distribution approaches a nearly one-hot form, effectively selecting the most suitable expert for each token. Following prior work (Nie et al., 2021), we dynamically reduce τ from 2.0 to 0.3 during training and use the hard mode during inference, selecting only one expert. Notably, only the global router does not conduct hard mode during inference, as we need experts from different expert groups to cooperate in accompaniment generation. The algorithm of Band-MOE is shown in Algorithm 1.

Moreover, to avoid overloading any individual expert and ensure balanced utilization, we incorporate a load-balancing loss for each expert group (Fedus et al., 2022). The balance loss $\mathcal{L}_{balance}$ is:

$$\mathcal{L}_{balance} = \alpha N \sum_{i=1}^{N} \left(\frac{1}{B} \sum_{h \in B} g(h)_i \right). \quad (14)$$

where B is the batch size, N is the number of experts, and α is a hyperparameter controlling the strength of the regularization, for which we use 0.1. This loss encourages a more uniform distribution of tokens across experts, improving training efficiency by preventing expert underutilization or overload. Thus, our routing strategy not only allows dynamic expert selection but also ensures that the computational load is evenly distributed across experts, reducing training time and improving the model performance of Band-MOE.

D.5 Classifier-free Guidance

During AccompBand training, we randomly replace the input text tokens with embedded empty strings at a probability of 0.2. The empty strings are processed through the text encoder to extract text tokens and are padded to a fixed length, like the original text prompts. For γ in Equation 2, a higher γ emphasizes the control of the text prompt, improving generation quality by making the outputs more aligned with the given conditions. In contrast, a lower γ allows for more diverse outputs by reducing the reliance on the text prompt, though this may result in lower relevance to the input prompt. In our major accompaniment generation experiments, we use $\gamma = 3$.

E LyricBand and MelodyBand

E.1 LyricBand

To enhance the customizability of our song generation system, we introduce LyricBand, a model designed to generate complete song lyrics based on arbitrary text prompts. Users can input parameters such as theme, emotion, genre, style, and specific keywords to generate fully personalized lyrics tailored to their preferences. To effectively train LyricBand, we leverage GPT-4o (Achiam et al., 2023) to extract prompts from a large corpus of existing song lyrics in our training data. These prompts encapsulate essential elements such as the thematic content, emotional tone, narrative perspective, rhyme scheme, and stylistic features of the songs. By extracting this rich set of attributes, we create a comprehensive dataset that pairs textual prompts with corresponding lyrics, enabling the model to learn the mapping between user inputs and desired lyrical outputs.

We employ QLoRA (Dettmers et al., 2024) for efficient fine-tuning of the well-performing open-source bilingual large language model Qwen-7B (Bai et al., 2023). By utilizing 4-bit quantization and low-rank adapters, QLoRA significantly reduces the computational resources required for fine-tuning while preserving the model's performance. This approach allows LyricBand to adapt effectively to the task of lyrics generation, maintaining high levels of customization and creativity across a diverse range of user prompts. In our experiments, we set LoRA $r = 32, \alpha = 16$. LyricBand demonstrates the capability to capture nuanced themes and emotions specified by users, generating lyrics that not only align with the given prompts but also exhibit coherent structure and artistic expression.

E.2 MelodyBand

Previous singing voice and song generation models often require users to provide music scores to achieve stable melodies (Zhiqing et al., 2024), lacking personalized customization of the melody. Inspired by symbolic music generation models (Dong et al., 2018; Ding et al., 2024), we introduce MelodyBand, an additional model where melody-related features like notes are generated from text descriptions in advance. By using notes as the representation of the melody, we can achieve more stable melody control. However, requiring users to provide music scores is impractical. Generating notes using natural language prompts can both

Algorithm 1 Pseudo-Code of Band-MOE Routing Strategy

Input: Input hidden representation h, vocal embedding z_v, text prompt embedding z_p, time step t
Output: Output with enhanced quality and control o_{final}

Initialize Gumbel-Softmax temperature τ, sample Gumbel noise ζ
for each time step t **do**
  Aligned MOE:
    Use Gumbel-Softmax for each token in the time channel to select an expert by z_v:
    $g_{\text{aligned}}(h) \leftarrow \text{GumbelSoftmax}(z_v \cdot W_{\text{aligned}} + \zeta)/\tau$
    Compute Aligned MOE output:
    $o_{\text{aligned}} \leftarrow \sum_i g_{\text{aligned},i} \cdot \text{Expert}_{i,\text{aligned}}(h)$
  Controlled MOE:
    Use Cross-Attention extracting style for alignment between z_p and h:
    $z_{sty} \leftarrow \text{CrossAttention}(h(Q), z_p(K), z_p(V))$
    Use Gumbel-Softmax for each token in the time channel to select an expert by z_{sty}:
    $g_{\text{controlled}}(h) \leftarrow \text{GumbelSoftmax}(z_{sty} \cdot W_{\text{controlled}} + \zeta)/\tau$
    Compute Controlled MOE output:
    $o_{\text{controlled}} \leftarrow \sum_i g_{\text{controlled},i} \cdot \text{Expert}_{i,\text{controlled}}(h)$
  Global Router:
    Use Gumbel-Softmax to compute global weights α_t and β_t:
    $g_{\text{global}}(h) \leftarrow \text{GumbelSoftmax}(embedding(t) \cdot W_{\text{global}} + \zeta)/\tau$
    $\alpha_t, \beta_t \leftarrow g_{\text{global}}(h)$
    Combine Aligned and Controlled MOE outputs:
    $o_{\text{combined}} \leftarrow \alpha_t \cdot o_{\text{aligned}} + \beta_t \cdot o_{\text{controlled}}$
  Acoustic MOE:
    Use Gumbel-Softmax to select an expert for each frequency channel:
    $g_{\text{acoustic}}(o_{\text{combined}}) \leftarrow \text{GumbelSoftmax}(o_{\text{combined}} \cdot W_{\text{acousitc}} + \zeta)/\tau$
    Compute Acoustic MOE output:
    $o_{\text{acoustic}} \leftarrow \sum_j g_{\text{acoustic},j} \cdot \text{Expert}_{j,\text{acoustic}}(o_{\text{combined}})$
end for
Return $o_{\text{final}} \leftarrow o_{\text{acoustic}}$ as the final routed output

ensure stable melodies and allow for flexible customization. For controllable melody generation, we construct artificial textual prompts to deliver melody-related information. Musical attributes like key, tempo, vocal range, and other information can be used as prompts for melody customization.

When users do not input music scores, as shown in Figure 6(b), MelodyBand takes the phonemes of the lyrics as content information and optional vocal prompts to extract timbre. It composes music for the lyrics and selects appropriate frequencies based on the timbre, using text prompts for style control. We employ a non-autoregressive transformer model to efficiently generate note pitches and durations simultaneously. The non-autoregressive transformer enables fast and high-quality generation, making it suitable for our multi-task song generation system.

With encoded phonemes and timbre, we inject text prompts through cross-attention transformers, allowing the model to integrate linguistic cues more

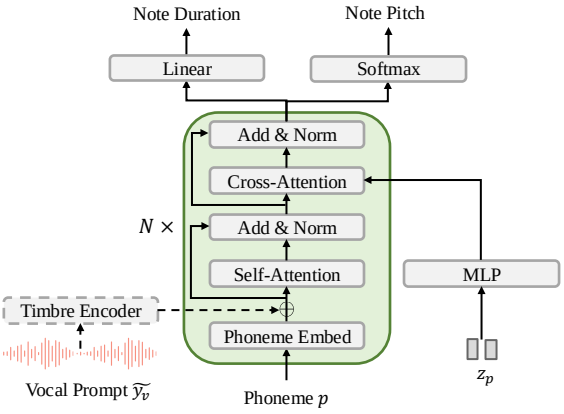


Figure 6: The architecture of MelodyBand.

effectively. Several heads are added to generate note pitches and durations. We pass each dimension of the stacked output through a softmax function to generate note pitches and through a linear layer to generate note durations. We train Melody-

Dataset	Type	Languages	Annotation	Duration (hours)
GTSinger (Zhang et al., 2024b)	vocal	Chinese & English	lyrics, notes,styles	29.6
M4Singer (Zhang et al., 2022a)	vocal	Chinese	lyrics, notes	29.8
OpenSinger (Huang et al., 2021)	vocal	Chinese	lyrics	83.5
LP-MusicCaps-MSD (Doh et al., 2023)	accomp	/	text prompt	213.6
web-crawled	song	Chinese, English	/	979.4

Table 9: Statistics of training datasets.

Band using cross-entropy loss for note pitches and an L2 loss for note durations. Let the true note pitch and duration for i-th phoneme be $n_p^{(i)}$ and $n_d^{(i)}$, and the GT note pitch and duration be $\hat{n}_p^{(i)}$ and $\hat{n}_d^{(i)}$, respectively. The cross-entropy loss $\mathcal{L}_{pitch}$ is:

$$\mathcal{L}_{pitch} = -\sum_{i=1}^{N}\sum_{k=1}^{K} \delta_{\hat{n_p}^{(i)},k} \log(P_k^{(i)}), \qquad (15)$$

where N is the length of phoneme sequence, K is number of pitch classes, $\delta_{\hat{n_p}^{(i)},k}$ is 1 if $\hat{n_p}^{(i)} = k$ and 0 otherwise. $P_k^{(i)}$ is the predicted probability of pitch k at time i. The L2 loss $\mathcal{L}_{duration}$ is:

$$\mathcal{L}_{duration} = \sum_{i=1}^{N} \left(n_d^{(i)} - \hat{n}_d^{(i)}\right)^2. \qquad (16)$$

Our MelodyBand employs 8 transformer layers, and 8 attention heads, the hidden size is 768, with 23.32M parameters in total.

F Dataset Analysis

We train our model using a combination of bilingual web-crawled song datasets and open-source singing datasets. Since there are no publicly available annotated song datasets, we collect 20k Chinese and English songs from well-known music websites. To expand data, we also use open-source singing datasets including GTSinger (Zhang et al., 2024b) (30 hours in Chinese and English), M4Singer (Zhang et al., 2022a) (30 hours in Chinese), and OpenSinger (Huang et al., 2021) (83 hours in Chinese). After processing and cleaning, we have about 1,000 hours of song data (about 80% in Chinese and 20% in English) and 1,150 hours of vocal data. For accompaniment generation, we use a filtered subset of LP-MusicCaps-MSD (Doh et al., 2023), resulting in a total size of around 1,200 hours. We use all open-source datasets under license CC BY-NC-SA 4.0. The statistics of the datasets are listed in 9.

For the web-crawled data, we use Ultimate Vocal Remover [2], an open-source music source separation tool, to perform the vocal-accompaniment separation. We utilize WhisperX (Bain et al., 2023) to automatically transcribe the demixed vocals, and Montreal Forced Aligner (MFA) (McAuliffe et al., 2017) is employed for phoneme and vocal alignment. After that, we filter the samples using Silero VAD (Team, 2021) to eliminate unvoiced clips. The samples are segmented into phrases with a maximum length of 20 seconds, resulting in an average segment duration of 12 seconds.

We utilize a music captioning model (Doh et al., 2023) to generate text prompts from the segmented song clips, and GPT-4o (Achiam et al., 2023) is used to separate music styles (such as genre, tone, and instrumentation) from vocal descriptions (such as emotion and gender). For singing styles, following style labels of GTSinger, we hire music experts to annotate all songs for the global singing method (e.g., pop or bel canto) and to label around 200 hours of segmented vocal clips for specific techniques used (e.g., mixed voice, falsetto, breathy, vibrato, glissando, and pharyngeal). We hire all music experts and annotators with musical backgrounds at a rate of $300 per hour. They have agreed to make their contributions available for research purposes. These annotations, along with the separated vocal descriptions, form the complete singing styles. For melody styles, we extract the key from the segmented demixed vocal clips using music21 [3], tempo and duration using librosa [1], and then use GPT-4o to combine these elements, generating natural language descriptions of vocal ranges based on the average pitch. For lyric styles, we process lyrics using GPT-4o to extract elements such as thematic content, emotional tone, narrative perspective, rhyme scheme, and stylistic features.

All styles are combined, along with various annotations, to form the final text prompts. During generation, we randomly omit certain elements or entire styles to enhance the model's generalization ability. We utilize ROSVOT (Li et al., 2024b) to obtain note sequences from the segmented demixed

[2] https://github.com/Anjok07/ultimatevocalremovergui

[3] https://github.com/cuthbertLab/music21

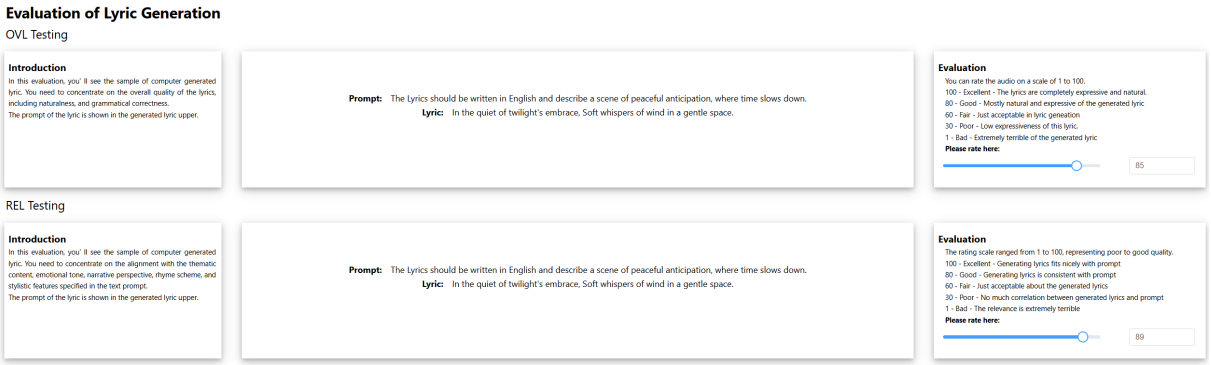


Figure 7: Screenshot of lyric evaluation.

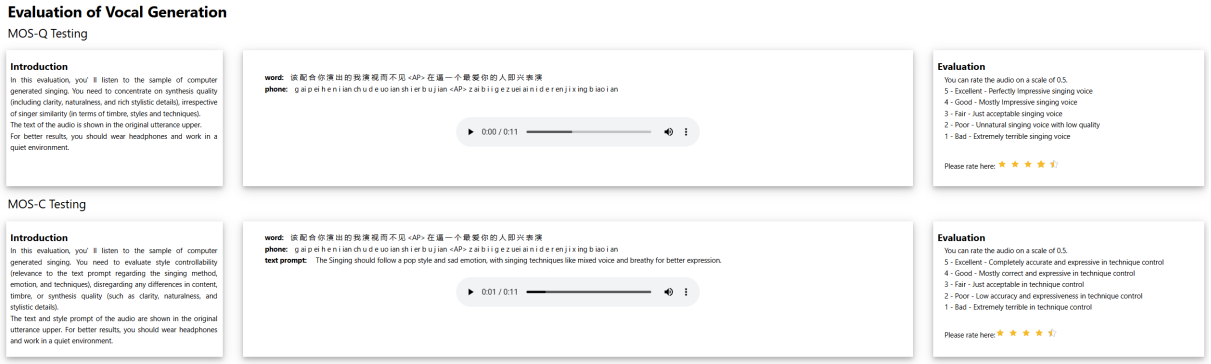


Figure 8: Screenshot of vocal evaluation.

vocal clips. For vocal and accompaniment data that lacks specific annotations, we use corresponding methods to complete the labeling process.

G Evaluation Metrics

G.1 Lyric and Melody Evaluation

For evaluating lyric generation, we randomly select 30 prompts and generate 30 sets of lyrics. Each set is evaluated by at least 15 raters for overall quality (OVL) and relevance to the prompt (REL) as subjective evaluation metrics. The rating scale ranged from 1 to 100, representing poor to good quality. OVL focused on the overall quality of the lyrics, including naturalness, and grammatical correctness, while REL assessed the alignment with the thematic content, emotional tone, narrative perspective, rhyme scheme, and stylistic features specified in the text prompt. All participants are fairly compensated for their time and effort at a rate of $12 per hour. They are also informed that the results will be used for scientific research purposes. The testing screenshot is shown in Figure 7.

In melody generation, multiple objective metrics are employed to evaluate controllability. We use the Krumhansl-Schmuckler algorithm to predict the potential key of the generated notes and report the average key accuracy (KA). If the Pearson correlation coefficient of the ground truth (GT) notes corresponding to the GT key is r, and the predicted MIDI corresponding to the GT key is $\hat{r}$, we define the key accuracy as $KA = \hat{r}/r$ (only valid if $r \neq 0$). We also compute the average absolute difference of average pitches (APD) and temporal duration (TD, in seconds). Moreover, following previous work (Sheng et al., 2021), we record pitch and duration distribution similarity (PD and DD). Specifically, we calculate the distribution (frequency histogram) of pitches and durations in notes and measure the distribution similarity between generated notes and ground truth notes:

$$\frac{1}{N_s}\sum_{i=1}^{N_s} OA(Dis_i, \hat{Dis}_i), \tag{17}$$

where Dis_i and $\hat{Dis}_i$ represent the pitch or duration distribution of the i-th generated and ground-truth song, respectively, N_s is the number of songs in the test set, and OA represents the average overlapped area. Melody distance (MD) is also computed with dynamic time warping (DTW) (Berndt and Clifford, 1994). To evaluate the pitch trend

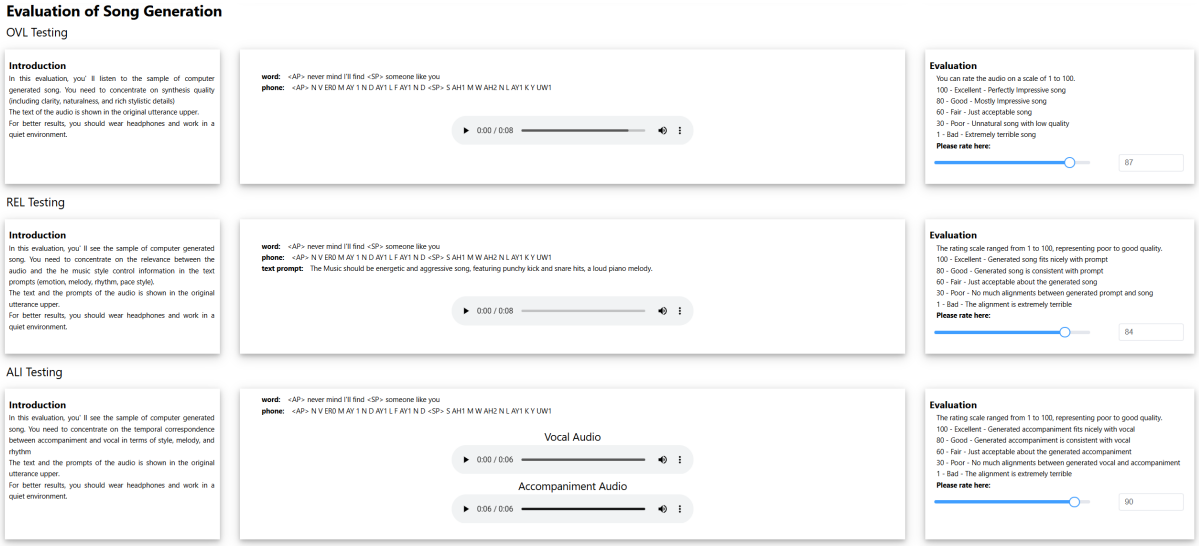


Figure 9: Screenshot of song evaluation.

of the melody, we spread out the notes into a time series of pitch according to the duration, with a granularity of 1/16 note. Each pitch is normalized by subtracting the average pitch of the entire sequence. To measure the similarity between generated and ground-truth time series with different lengths, we use DTW to compute their distance.

G.2 Vocal Evaluation

For vocal generation, we randomly select 30 pairs of sentences from our test set for subjective evaluation. Each pair consists of a ground truth (GT) and a synthesized vocal, each listened to by at least 15 professional listeners. For MOS-Q evaluations, these listeners are instructed to focus on synthesis quality (including clarity, naturalness, and richness of stylistic details) without considering the style control relevance to text prompts. For MOS-C, the listeners are informed to evaluate style controllability (relevance to the text prompt regarding the singing method, emotion, and techniques), disregarding any differences in content, timbre, or synthesis quality (such as clarity, naturalness, and stylistic details). In both MOS-Q and MOS-C evaluations, listeners are asked to grade various vocal samples on a Likert scale from 1 to 5. For fairness, all samples are resampled to 24kHZ. The screenshots of instructions are shown in Figure 8.

We employ F0 Frame Error (FFE) to evaluate the test set's synthesis quality objectively. FFE combines metrics for voicing decision error and F0 error, capturing essential synthesis quality information. For comparison with the FFE reported in the MelodyLM paper, we resample all audio to 24kHz.

For singing style transfer, subjective evaluation is conducted using pairs of audio, where each pair includes a prompt vocal and a synthesized vocal. During MOS-S evaluations, listeners are asked to assess singer similarity in terms of timbre and personalized styles to the vocal prompt, disregarding any differences in content or synthesis quality.

To objectively evaluate timbre similarity, we employ Cosine Similarity (Cos). Cos measures the resemblance in singer identity between the synthesized vocal and the vocal prompt by computing the average cosine similarity between the embeddings extracted from the synthesized voices and the vocal prompt, thus providing an objective indication of singer similarity performance. Specifically, we use a voice encoder [4] to extract singer embeddings.

In all MOS-Q, MOS-S, and MOS-C evaluations, listeners are requested to grade the vocal samples on a Likert scale ranging from 1 to 5. All participants are fairly compensated for their time and effort. We compensate participants at a rate of $12 per hour. Participants are informed that the results will be used for scientific research.

G.3 Song Evaluation

For the subjective evaluation of song generation, we randomly select 30 audio samples from our test set. Each sample is listened to by at least 15 raters. Following previous work (Copet et al., 2024; Zhiqing et al., 2024), we ask human raters to evaluate three aspects of the audio samples: (i) overall quality (OVL), (ii) relevance to the text prompts

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

(REL), and (iii) alignment with the vocal (ALI). For the overall quality test, raters are asked to rate the perceptual quality of the provided samples. For the text relevance test, raters evaluate how well the audio matches the music style control information in the text prompts. For the alignment with the vocal test, raters focus on the temporal correspondence between accompaniment and vocal in terms of style, melody, and rhythm. Ratings are given on a scale from 1 to 100.

All participants are fairly compensated for their time and effort, with a rate of $12 per hour. Participants are informed that the results will be used for scientific research. For fairness, all samples are resampled to 24kHZ and normalized to -23dB LUFS (Series, 2011). The screenshots of instructions in the song generation task are shown in Figure 9.

For the objective evaluation, we use Frechet Audio Distance (FAD), Kullback-Leibler Divergence (KLD), and the CLAP score (CLAP). We report the FAD (Kilgour et al., 2018) using the official implementation in TensorFlow with the VGGish model [5]. A low FAD score indicates that the generated audio is plausible. Following previous work (Copet et al., 2024), we compute the KL-divergence over the probabilities of the labels between the GT and the generated music. Finally, the CLAP score (Wu et al., 2023) is computed between the track description and the generated audio to quantify audio-text alignment, using the pre-trained CLAP model [6].

H Baseline Models

For lyric generation, we employ the fine-tuned Qwen-7B model as our lyric generation component and use its original (non-fine-tuned) version as the baseline. For melody generation, we compare against SongMASS (Sheng et al., 2021) and the MIDI component of MelodyLM (Li et al., 2024a), both of which are capable of generating MIDI sequences based on Transformer architectures.

For vocal generation, we compare with VISinger2 (Zhang et al., 2022b), a traditional high-fidelity SVS model, StyleSinger (Zhang et al., 2024a), the current state-of-the-art zero-shot SVS model with style transfer capabilities. We incorporate our text encoder and style alignment modules into the open-source VISinger2 and StyleSinger implementations to enable style control. These two models represent well-performing open-source SVS baselines: one follows the traditional SVS paradigm, while the other supports zero-shot style transfer and style modeling. Meanwhile, we also compare with vocal parts of Melodist (Zhiqing et al., 2024) and MelodyLM. They both leverage Transformer models for vocal generation, allowing for direct comparisons like other SVS models. It should be noted that Melodist lacks zero-shot capabilities and style transfer features. Similarly, Melodist and MelodyLM cannot perform natural language prompt–based style control.

For song generation, we compare with Melodist and MelodyLM, two representative text-to-song models. They publicly report datasets of a similar scale to ours and provide comprehensive subjective and objective evaluation metrics for song generation, thus enabling fair comparisons. We do not employ the recently proposed Seed-Music (Bai et al., 2024) as a baseline model because it was introduced very recently, does not provide open-source code, and lacks details on dataset type and size, as well as comprehensive objective and subjective evaluation results. Additionally, we did not compare our work with SongCreator (Lei et al., 2024) because its task design does not support style control via natural language prompts, excludes melody information, lacks open-source code, and differs significantly in both task scope and the types of evaluation metrics reported. Consequently, we were unable to conduct a meaningful comparison. Meanwhile, we do not include pure music generation models as baselines, since these models are typically closed-source, not adapted for vocal alignment, and differ significantly in both data usage and task definition. For Melodist and MelodyLM, we rely on their papers and available demos for evaluation. For the other models, we use the corresponding open-source codes.

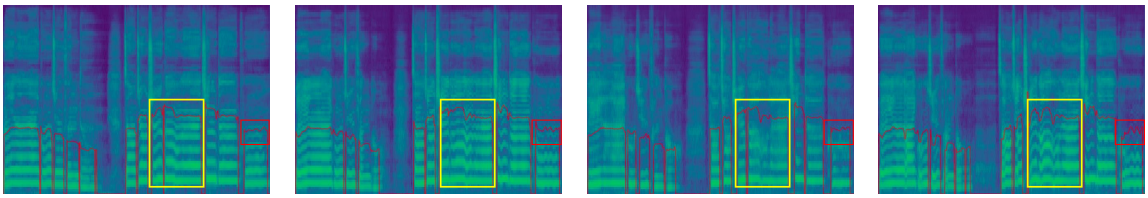
I Multi-Task Experiments

I.1 Vocal Generation

In Figure 10, we compare the mel-spectrograms of VocalBand with different singing styles specified in the text prompt. Figure 10 (a) represents the GT vocal, where the mel-spectrogram within the yellow box is relatively uniform, indicating a stable vocal performance, while the F0 contour in the red box is smooth. In contrast, Figure 10 (b) does not specify singing styles, allowing the free use of techniques to enhance expressiveness, as seen

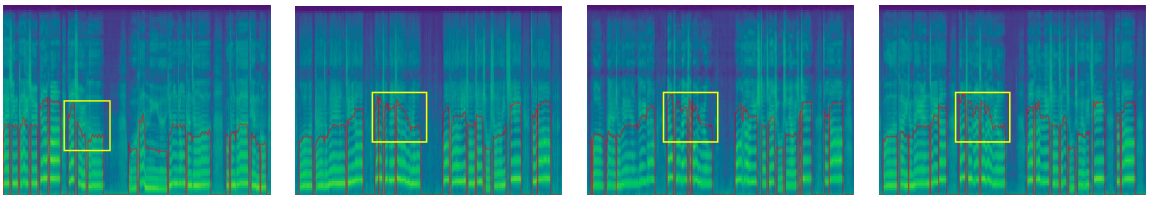
[5] https://github.com/google-research/google-research/tree/master/frechet_audio_distance

[6] https://github.com/LAION-AI/CLAP



(a) GT (b) w/o singing style (c) breathy technique (d) bubble technique

Figure 10: Visualization of the mel-spectrogram results generated by VocalBand under different singing styles in the text prompt. The red box contains the fundamental pitch, and the yellow box contains the details of harmonics.



(a) Vocal Prompt (b) VISinger2 (c) StyleSinger (d) VocalBand

Figure 11: Visualization of the mel-spectrogram results generated by VocalBand for singing style transfer. The yellow box contains the fundamental pitch.

by the significant pitch oscillations in the red box, characteristic of vibrato. In Figure 10 (c), representing the breathy technique, the mel energy in the yellow box shows a significant drop in high-frequency energy, consistent with the softer, airier vocal timbre of breathy singing. Finally, Figure 10 (d) illustrates the bubble technique, where the yellow box displays pronounced low-frequency energy with more exaggerated vertical modulations. The red box shows a distinctive pitch fluctuation pattern, characterized by slower, larger oscillations, indicative of the unique vocal fold vibrations in this technique. These results demonstrate that VocalBand can achieve diverse and highly expressive control over the same content based on the different singing styles specified in the text prompt.

I.2 Singing Style Transfer

In Figure 11, we compare the performance of VocalBand and baseline models on singing style transfer. It can be observed that VocalBand excels at capturing the intricate nuances of the prompt style. The pitch curve generated by VocalBand displays a greater range of variations and finer details, closely resembling the prompt style. In the yellow boxes, it is evident that VocalBand captures nuances in pronunciation and articulation skills similar to the vocal prompt. In contrast, the curves generated by other methods appear relatively flat, lacking distinctions in singing styles. Moreover, the mel-spectrograms generated by VocalBand exhibit superior quality, showcasing rich details in frequency bins between adjacent harmonics and high-frequency components. In contrast, the mel-spectrograms produced by other methods demonstrate lower quality and a lack of intricate details.

I.3 Music Style Transfer

For music style transfer, AccompBand uses the noisy prompt accompaniment $\tilde{y_a}$ with a time step 0.5 instead of Gaussian noise ϵ and sums it with the target vocal y_v, enabling the model to learn the style from the retained components of the prompt accompaniment. Thus, we do not need a text prompt to control the music style. We use ALI-A for subjective evaluation of the style similarity to the prompt accompaniment. As shown in Table 10, we achieve good style similarity with minimal changes in quality. This demonstrates that VersBand, leveraging AccompBand's flow matching mechanism, can also effectively perform the music style transfer task.

I.4 Vocal-to-Song Generation

We can directly input GT vocals for the vocal-to-song generation task. We compare our method with MelodyLM, which also generates songs from GT vocals. We use the objective metrics reported in their papers and subjectively evaluate the demos on their demo pages. As shown in Table 11, it is evident that with GT vocal input, VersBand achieves improved quality and better alignment with the

Methods	FAD ↓	KLD ↓	OVL ↑	ALI-A ↑
VersBand (w/o prompt)	3.01	1.27	**87.92±1.73**	-
VersBand	**3.02**	**1.26**	87.34±1.28	**80.24±1.57**

Table 10: Results of music style transfer. Prompt means prompt accompaniment.

Methods	FAD ↓	KLD ↓	OVL ↑	ALI ↑
VersBand (w/o GT)	3.01	1.27	87.92±1.73	80.51±1.66
MelodyLM	3.13	1.31	84.67±1.23	75.19±0.82
VersBand	**2.65**	**1.19**	**90.17±1.55**	**83.54±1.32**

Table 11: Results of vocal-to-song generation. GT means GT vocal.

Methods	MOS-Q↑	FEE↓
VocalBand (w/o GT)	4.04±0.08	0.07
StyleSinger	3.79±0.10	0.09
VocalBand	**3.87±0.05**	**0.08**

Table 12: Results of accompaniment-to-song generation. GT means GT accompaniment.

vocals compared to song generation without GT vocal input, and it outperforms MelodyLM. This is because the GT vocal provides a more accurate style, melody, and rhythm, better matching the target accompaniment. It demonstrates that VersBand effectively utilizes AccompBand's excellent vocal alignment mechanisms of Aligned MOE, to accomplish the Vocal-to-Song Generation task.

I.5 Accompaniment-to-Song Generation

We use ROSVOT (Li et al., 2024b) to extract notes from the accompaniment to guide VocalBand for vocal generation. The extracted notes are also provided to StyleSinger, which can similarly utilize notes, as a baseline model. As shown in Table 12, it is evident that the quality decreases when using GT accompaniment instead of music scores, as the notes from the accompaniment are not aligned with the vocal notes, primarily due to differences in their characteristics. Vocals often involve techniques and emotional expression, with pauses between words. At the same time, accompaniments are more complex, involving multiple instruments and rarely pausing, leading to discrepancies in timing and pitch between the vocal and accompaniment notes. However, VocalBand still outperforms StyleSinger and achieves satisfactory results. This demonstrates that VersBand can leverage the user's preferred GT accompaniment for vocal pairing, with VocalBand exhibiting excellent rhythm and melody control by decoupling content.

J Ablation Study

J.1 Experiments on Text Encoder

For the text encoder, following previous work (Zhiqing et al., 2024), we test FLAN-T5-large (Chung et al., 2024), BERT-large (Devlin et al., 2018), and the text encoder of CLAP (Elizalde et al., 2023). Table 13 shows that we test VersBand without inputting lyrics or music scores. It can be seen that the T5 text encoder outperforms the other two text encoders in both quality and relevance, but has only a slight advantage over BERT, which is likely due to its larger parameter count and multi-task capability.

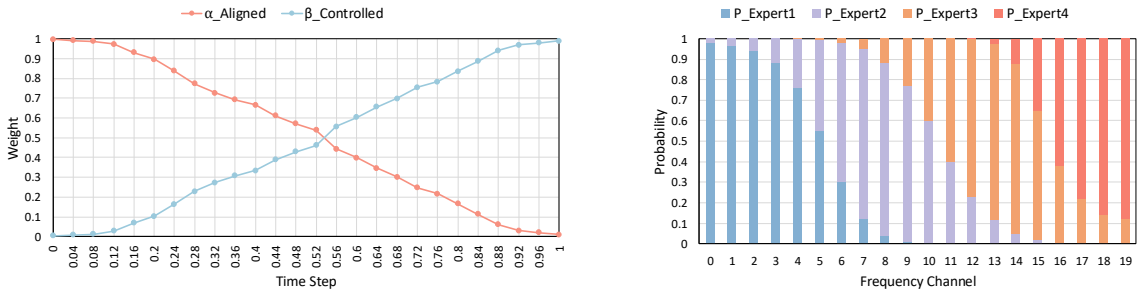
J.2 Experiments on MOE

To demonstrate the effectiveness of our MOE, we conducted experiments on the final routing behavior. As shown in Figure 12 (a), we can observe that our global routing behaves as expected. As the noise level decreases, the weighting of outputs from Aligned MOE and Controlled MOE changes accordingly: 1) At early time steps (near 0), where the hidden representation h is highly noisy, the network prioritizes matching with the vocal for coherent reconstruction, thus the weight of the Aligned MOE is higher. 2) At later time steps (near 1), where h has been largely reconstructed, the network focuses more on refining stylistic details, relying heavily on text prompts, thus the weight of the Controlled MOE is higher.

As shown in Figure 12 (b), the Acoustic MOE also behaves as expected by assigning different experts to different channels. We encode the mel-spectrogram into 20 dimensions through the acomp encoder, resulting in 20 channels and selecting experts for each channel. We perform a statis-

Methods	FAD ↓	KLD ↓	CLAP ↑	OVL ↑	REL ↑
VersBand (T5)	**3.01**	**1.27**	**0.58**	**87.92±1.73**	**88.03±0.59**
VersBand (CLAP)	3.31	1.34	0.41	85.36±1.57	86.03±1.39
VersBand (BERT)	3.12	1.29	0.49	87.02±0.84	87.21±0.83

Table 13: Results of ablation study on different text encoders.



(a) Global Routing (b) Acousitc Routing

Figure 12: The statistics of global routing and acoustic routing in Band-MOE.

tical analysis of the softmax output probabilities before expert selection. 1) Expert 1 focuses on channels 0 to 7, which include instruments that provide foundational rhythm and depth, such as bass guitars, kick drums, low-frequency percussion, and the lower registers of piano and organ. 2) Expert 2 specializes in channels 4 to 12, capturing the richness of rhythm guitars, mid-range piano notes, and various percussion instruments that contribute to the fullness and body of the music. 3) Expert 3 targets channels 9 to 16, encompassing lead guitars, higher piano octaves, string instruments, and brass instruments. This allows the model to capture melodic elements and intricate harmonics that enhance the expressiveness of the accompaniment. 4) Expert 4 is assigned to channels 14 to 19, focusing on cymbals, hi-hats, flutes, and high-frequency string overtones that contribute to the brightness and airiness of the music.

K MIDI and F0

The pitch predicted by VocalBand refers to the F0 of the vocal, while MelodyBand predicts the MIDI music score. These two representations are fundamentally different in terms of both their structure and level of granularity.

F0 (Fundamental Frequency) refers to the lowest frequency of a periodic waveform, typically representing the pitch of a sound. In vocal music, it corresponds to the pitch produced by the singer's voice. f0 is continuous and fine-grained, capturing subtle pitch variations that occur naturally during singing. This includes slight pitch bends, vibrato, and dynamic vocal expressions. The vocal f0 prediction in VocalBand focuses on capturing these nuanced and continuous pitch variations that define vocal performances.

MIDI (Musical Instrument Digital Interface) is a digital representation of music that encodes musical notes, chords, velocities, and other musical events in a discrete, symbolic format. MIDI defines music in terms of note numbers (representing pitch) and timing information (e.g., note onset and duration). Unlike f0, MIDI is typically coarser in its resolution. It represents pitch as discrete notes (e.g., C4, D5), with fixed intervals, meaning it does not capture the smooth pitch fluctuations found in vocal performances or continuous instruments. The MIDI music score predicted by MelodyBand works with these discrete representations, focusing more on the overall structure of the music (i.e., sequence of notes and timing) rather than the fine details of pitch fluctuations. In summary, f0 is a detailed, continuous signal that reflects the pitch of vocal sounds, whereas MIDI is a more abstract, discrete representation of musical notes with a lower level of granularity. These differences illustrate the challenge and complexity of predicting both types of information, each serving a distinct purpose in music generation and synthesis.

Modeling vocal MIDI as an intermediate modality for supervision can significantly improve the synthesis performance of models. Directly predicting f0 in VocalBand would complicate the modeling of phoneme duration and the f0 itself, leading to

poorer results and making it harder for the model to perform core synthesis tasks. Additionally, singing voice synthesis models typically require user input of music scores for coarse control. To maintain consistency within our model, we opted to include MIDI prediction.

L Reproducibility Statement

We have implemented several measures to ensure reproducibility: 1) We provide very detailed explanations of each module in our Appendix B, C, D, and E. We will also release the code after the paper is accepted. 2) We offer hyperparameters and experimental configurations for each model in Appendix B.1, C.1, D.1, E.1, and E.2. 3) The data processing steps and the open-source tools we used are described in detail in F. Since our datasets consist of open-source singing voices and songs collected from the internet, we will provide all web links and corresponding text prompt annotations after the paper is accepted. 4) All evaluation metrics are thoroughly described in Appendix G.