SongSong: A Time Phonograph for Chinese SongCi Music from Thousand of Years Away

Jiliang Hu[1,†], Jiajia Li[3,†], Ziyi Pan[1], Chong Chen[4], Zuchao Li[2,*], Ping Wang[3,*], Lefei Zhang[2]

[1]Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China,
[2]School of Computer Science, Wuhan University, Wuhan, China,
[3]School of Information Management, Wuhan University, Wuhan, China,
[4]ShenYang Conservatory Of Music, ShenYang, China.

Abstract

Recently, there have been significant advancements in music generation. However, existing models primarily focus on creating modern pop songs, making it challenging to produce ancient music with distinct rhythms and styles, such as ancient Chinese SongCi. In this paper, we introduce SongSong, the first music generation model capable of restoring Chinese SongCi to our knowledge. Our model first predicts the melody from the input SongCi, then separately generates the singing voice and accompaniment based on that melody, and finally combines all elements to create the final piece of music. Additionally, to address the lack of ancient music datasets, we create OpenSongSong, a comprehensive dataset of ancient Chinese SongCi music, featuring 29.9 hours of compositions by various renowned SongCi music masters. To assess SongSong's proficiency in performing SongCi, we randomly select 85 SongCi sentences that were not part of the training set for evaluation against SongSong and music generation platforms such as Suno and SkyMusic. The subjective and objective outcomes indicate that our proposed model achieves leading performance in generating high-quality SongCi music.

Project — https://zcli-charlie.github.io/songsong/

Introduction

SongCi, as a brilliant pearl in the history of Chinese poetry, is an important carrier of excellent Chinese culture and carries the spiritual pursuit of the Chinese nation for thousands of years. As shown in Figure 1, these ancient poems that have been passed down to today can not only be recited, but also sung as lyrics. Unfortunately, due to historical changes and the loss of ancient music scores, the original musical forms of SongCi have largely disappeared and buried in the torrent of history.

In recent years, music understanding (Li et al. 2024; Tian et al. 2024) and generation have seen considerable advancements. In terms of music generation, numerous studies have focused on generating singing voices (Liu et al. 2022a; Chen et al. 2020) and melodies (Yu, Srivastava, and Canales 2021; Zhang et al. 2022a; Lv et al. 2022). Deep learning techniques can potentially restore SongCi music, but there is presently no open-source dataset available for ancient Chinese SongCi. Existing Chinese music datasets, like M4Singer (Zhang et al. 2022b) and OpenCPOP (Wang et al. 2022), are limited to pop songs. Currently, most models capable of generating complete music with singing and accompaniment are commercial products, such as Suno and SkyMusic. These models leverage the powerful generative abilities of large language models (LLMs) (Zhao et al. 2023) to create full audio based on user-provided text prompts and lyrics. Additionally, users can input audio as a style reference for the model to generate music accordingly. However, the generation process in these models is often poorly controllable and highly random. Furthermore, the training data for these models primarily consists of contemporary popular songs, resulting in generated music that predominantly reflects mainstream styles instead of ancient music styles. Although Shan et al. (2023) previously proposed an ancient Chinese poem-to-song system, due to the insufficiency of structure superiority and the lack of ancient music corpus, the music generated by this system is still closer to modern music.

Figure 1: A piece of ancient Chinese SongCi music recorded using Chinese Gongche notation, which is vertically arranged and uses special Chinese symbols as musical notes. The blue box indicates the SongCi, and the red box indicates the notes.

In this study, we introduce SongSong, a music generation model that can perform Chinese SongCi. The model utilizes the auto-regressive Transformer and the diffusion

†These authors contribute equally to this work.
*Corresponding authors.

technology. It takes a SongCi poem as input, first creating a melody that corresponds to the poem. Next, it synthesizes the singing voice and generates accompaniment based on the music score derived from the melody. Finally, the singing voice, melody, and accompaniment are combined to create the final piece of music. To tackle the scarcity of ancient music datasets and aid in the restoration of ancient music, we present OpenSongSong, a comprehensive dataset of ancient Chinese SongCi music, totaling 29.9 hours of SongCi music. We conduct experiments comparing SongSong with Suno and SkyMusic, evaluating the results from both subjective and objective viewpoints. The findings indicate that our proposed model can generate high-quality SongCi music that aligns with the traditional style of SongCi. The key contributions of this work can be summarized as follows:

(1) We propose SongSong, the first music generation model that can perform SongCi music to our knowledge.

(2) We have developed a comprehensive SongCi music dataset to address the shortage of publicly available ancient Chinese music datasets and to support the restoration of SongCi music.

(3) Experimental results demonstrate that the proposed model achieves state-of-the-art performance in generating SongCi music.

Related Work

Singing Voice Synthesis Singing Voice Synthesis (SVS) is an attracting technique that uses musical score information (such as lyrics, tempo, pitch, etc.) to generate natural and expressive singing voices (Yamamoto, Song, and Kim 2020). In recent years, deep learning has brought revolutionary progress in the field of SVS, achieving significant improvements over traditional methods. Early deep learning-based systems use Feedforward Neural Networks (FFNN) (Nishimura et al. 2016), which outperforms HMM-based (Rabiner 1989) systems by predicting acoustic features directly from musical scores. Further developments introduce Long Short-Term Memory (LSTM) (Kim et al. 2018) and Convolutional Neural Networks (CNNs) (Nakamura et al. 2019, 2020), which enhances the ability to model long-term dependencies and acoustic features in singing voices. Generative Adversarial Networks (GAN) (Hono et al. 2019; Chandna et al. 2019) is also integrated into SVS systems to mitigate the over-smoothing problem, leading to more natural and expressive singing voice synthesis. Moreover, state-of-the-art deep learning architectures such as Transformer-based models (Vaswani 2017) like XiaoiceSingv (Lu et al. 2020), GAN-based models like HifiSinger (Chen et al. 2020), and diffusion-based models (Ho, Jain, and Abbeel 2020) like DiffSinger (Liu et al. 2022a), have further improved the quality of SVS.

Melody Generation Lyric-to-melody generation is a key task in automatic composition, which involves generating a melody that matches a given lyric. It usually uses an end-to-end (E2E) model to generate melodies directly from lyrics. Bao et al. (2019) and Yu, Srivastava, and Canales (2021) use sequence-to-sequence models to generate melodies from lyrics. However, these E2E models require a large amount of paired lyrics and melody data, and obtaining a sufficient amount of paired data is both difficult and costly. To address this problem, Sheng et al. (2021) avoid the reliance on paired data by training lyric-to-lyric and melody-to-melody models separately and interacting them in subsequent stages. However, since unpaired data is not fully utilized in learning the correlation between lyrics and melody, the consistency of lyrics and melody features cannot be ensured. Ju et al. (2022) propose TeleMelody, a generative model with two-stage: lyrics-to-template and template-to-melody. The template bridges the gap between lyrics and melody, promotes better alignment of features, and improves the controllability of generated melody.

Method

Figure 3 illustrates the structure of our model, SongSong, which is composed of four modules. It takes SongCi poetry as input and produces SongCi music audio as output. In the first stage, the lyric-to-rhythm module predicts the rhythm, which includes specific tonality and chord details, for each word in the poetry. In the second stage, the rhythm is processed by the rhythm-to-melody module to determine the corresponding note for each word. Next, we transform the sequence of notes into a melody MIDI file along with a config file. This config file contains information about the phonemes of the lyrics, the notes, and the fundamental frequency (f0) for the audio to be created, which is used to generate the singing voice audio. The melody audio is then utilized by the accompaniment generation module to create accompaniment played by other instruments, enhancing the overall richness of the audio. Ultimately, the complete SongCi music audio is produced by merging the singing voice audio, melody audio, and accompaniment audio.

Lyric To Melody

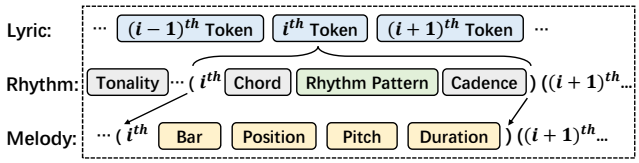

Figure 2: The relationship between lyric, rhythm, and melody.

The differences between lyrics and melody are quite pronounced. To create a model that can directly link lyrics to melody, a substantial amount of training data is necessary. However, since there is currently no extensive dataset for Chinese SongCi music, we utilize a two-stage method involving rhythm transitions to transform lyrics into melody.

Figure 2 illustrates the connections between lyrics, rhythm, and melody. The format of rhythm sequence is inspired by Ju et al. (2022) and encompasses tonality, chords, rhythm patterns, and cadences. Tonality consists of a scale and a root note, with each piece having only one tonic, meaning each rhythm also contains just one tonality. However, a rhythm can include multiple chords, rhythm patterns,

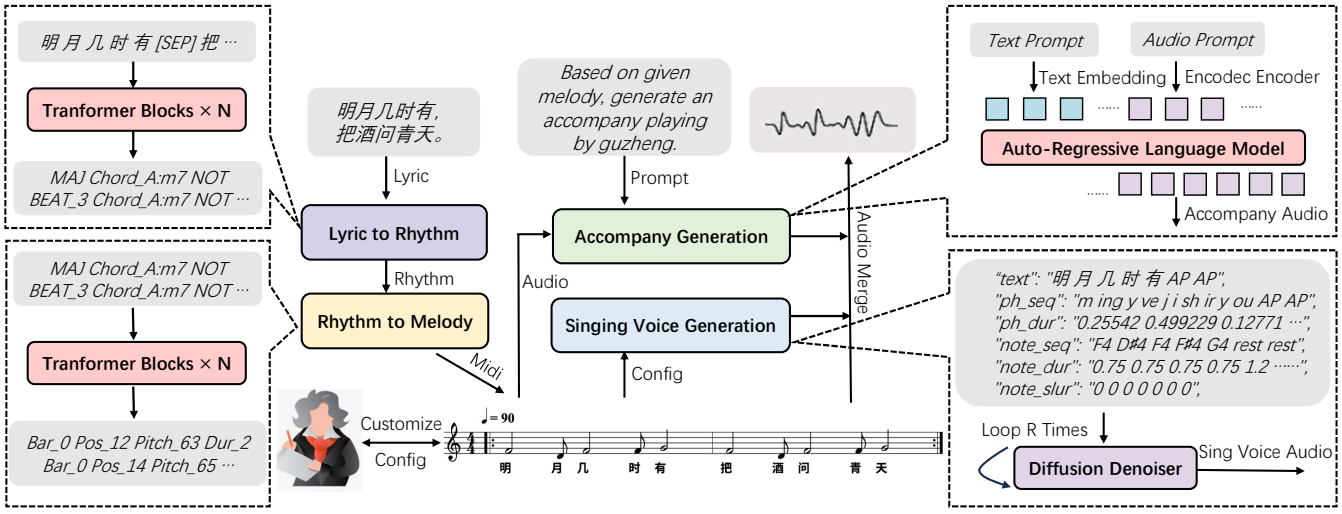

Figure 3: The structure of our proposed model, SongSong. The English meaning of input SongCi is “How rare the moon,so round and clear! With cup in hand,I ask of the blue sky.”

and cadences corresponding to the lyrics. Chords are collections of sounds with specific interval relationships, determined by a chosen chord progression. Cadences indicate the conclusion of a melodic section and can be classified as ”no cadence,” ”half cadence,” or ”authentic cadence,” depending on whether the associated lyric is a regular token, a pause token, or a final pause token. The rhythm pattern is derived from the lyric-to-rhythm module which is consist of stacked Transformer blocks, essentially representing the beat that aligns with the lyrics. Let X denote the lyric sequence, which has a length of n, and Y represent the beat sequence, which is of the same length.

$$Y = \text{Softmax}(\text{Transformer}(X))$$

The rhythm-to-melody module has the same structure as lyric-to-rhythm module. It takes the rhythm as input and produces a note sequence that corresponds to the lyrics. Each note is represented as a quadruple that includes the bar, position, pitch, and duration. The bar and position are used to determine the initial sound time of the note, pitch indicates the note’s pitch, and duration specifies how long the note sounds. Let Z represent the note sequence, which has a length of n.

$$Z = \text{Softmax}(\text{Transformer}(Y))$$

The training goal of these two modules is to minimize the negative log-likelihood on the lyric-to-rhythm data $(X_i Y_i)$ or the rhythm-to-melody data (Y_i, Z_i), which serves as the loss function. For instance, the optimization goal for the lyric-to-rhythm module is as follows.

$$\mathcal{L}_{lt} = -\mathbb{E}_{(X,Y)} log P(Y_i|X_i)$$

Melody To Singing Voice

Our singing voice generation model utilizes phonetic units, meaning the input consists of a phonetic sequence P instead of a lyric sequence X. The length of P corresponds to the total number of phonetic units m. However, having just P is insufficient; we also need to determine the duration of each phoneme, represented as Q, which also has a length of m and cannot be calculated using rule-based methods. Furthermore, the singing voice module requires melody information to ensure the lyrics are sung at the correct pitch. In the previous section, we derived a sequence of note quadruplets Y, which includes the note sequence U and the note duration sequence V. These are matched one-to-one with each lyric, resulting in both U and V having a length of n. To produce a complete song, we also need the fundamental frequency information $F0$, which has a length equal to the number of frames T in the generated audio. $F0$ must also be inferred using a machine learning model. Figure 4 illustrates the detailed design of the singing voice generation module. The variance encoder is a speech acoustic encoder based on FastSpeech2 (Ren et al. 2020), consisting of Transformer layers. It takes P as input and produces the phonetic acoustic feature H^{VE} from the last hidden layer.

$$H^{VE} = \text{TransformerEncoder}(P)$$

We utilize FastSpeech2’s duration predictor to estimate the duration of each phoneme. This predictor’s core component is a convolutional network (LeCun et al. 1998) made up of two layers of one-dimensional convolutions. It takes the note information U, V, and the phonetic acoustic feature H^{VE} as input.

$$Q = \text{Convolution}(U \oplus V \oplus H^{VE})$$

During training, we optimize the predictor by calculating the $L2$ norm of phoneme duration.

$$\mathcal{L}_{dp} = -\mathbb{E}_{(Q)}(Q_i - \hat{Q}_i)^2$$

The $F0$ predictor is based on a diffusion model (Nichol and Dhariwal 2021). It first determines the total number of

frames T for the generated audio by adding up the phoneme durations Q and creates a mapping matrix that connects phonemes to the mel spectrum. This mapping matrix is then used to extend the length of the acoustic features from m to T. Finally, $H^{VE}_{1:T}$ is used as a condition and conduct $R^{'}$ rounds of denoising from the original noise $F0_{R'}$ to reconstruct the pitch sequence. The $F0$ predictor is implemented using Wavenet (Oord et al. 2016), which is a fully probabilistic and auto-regressive model composed of causal convolutional layers. During training, the $F0$ predictor employs the $L2$ norm as its loss function.

$$F0 = \text{Diffusion}(F0_{R'}, H^{VE}_{1:T}, R^{'})$$

After acquiring the phoneme duration Q and $F0$, we will generate the Mel spectrum M for the singing voice, which consists of T frames. Initially, we will use an acoustic encoder to derive the acoustic representation H^{AE}. The inputs for the acoustic encoder include the phoneme P, phoneme duration Q, and $F0$, indicating that H^{AE} captures essential information about the singing technique. The basic setup of the acoustic encoder mirrors that of the variance encoder.

$$H^{AE} = \text{TransformerEncoder}(P \oplus Q \oplus F0)$$

Subsequently, we will feed the vocal acoustic features H^{AE} as a condition into a diffusion model to reconstruct the Mel spectrum of the singing voice. The standard inference process for the diffusion model is outlined, where M_R represents the initial Gaussian noise and R denotes the number of denoising iterations.

$$M = \text{Diffusion}(M_R, H^{AE}, R)$$

To enhance the quality of the generated audio and speed up the inference of the diffusion model, we implement the shallow diffusion mechanism introduced by DiffSinger. This mechanism firstly employs a ConvNeXt-based auxiliary decoder (Liu et al. 2022b), which consists of depth-wise convolutional blocks, to infer the spectrum $\tilde{M}_K$ at the K-th denoising step.

$$\tilde{M}_K = \text{Convolution}(H^{AE})$$

Then, instead of conducting R rounds of denoising from the original noise, the denoiser performs K rounds starting from $\tilde{M}_K$.

$$M = \text{Diffusion}(\tilde{M}_K, H^{AE}, K)$$

The denoiser's base model is identical to the $F0$ predictor. During the training phase, we enhance the auxiliary decoder by computing the $L1$ norm of the Mel spectrum, while the denoiser utilizes the $L2$ norm.

$$\mathcal{L}_{aux} = -\mathbb{E}_{(M)}|M_i - \hat{M}_i|$$

The config file created in this section defines the singing style for the voice generation model and can also be used to alter the melody MIDI produced by the rhythm-to-melody module. Since the music generated by the model may not appeal to everyone, we believe that a music generation model allowing for flexible control over the output content is more universally applicable. Consequently, the config file generated by SongSong is accessible to users. If users are dissatisfied with the melody produced, they can modify the config file independently to adjust both the generated melody audio and voiceover audio at the same time.

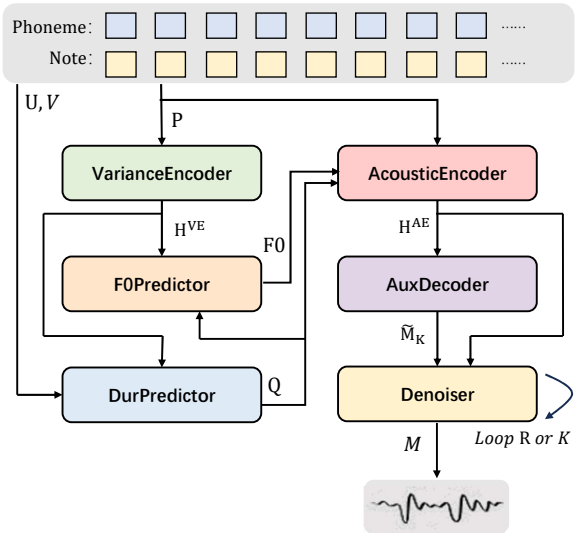

Figure 4: The design of the singing voice generation module.

Melody To Accompany

After completing the previous two sections, we have acquired the vocal audio and melody audio, which can be merged to produce a full music track. To enrich the final music output, we can utilize an additional generative model to create accompaniments played by other instruments.

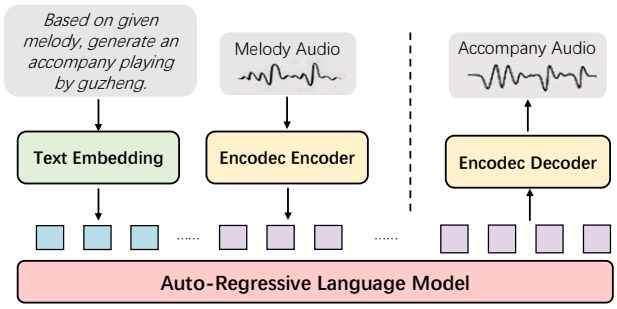

Figure 5: The specific architecture of adopted accompany generation module.

Following the approach outlined by Copet et al. (2024), we implement an architecture that discretizes acoustic features and generates discrete acoustic tokens through an autor-egressive language model for accompaniment generation. Figure 5 illustrates the specific structure of the accompaniment generation module. The input for this module consists of a text prompt and an audio prompt, with the audio prompt being the melody audio. The continuous audio features are quantized using the encoder from Encodec

Dataset	Style	Hours	Singers	Phoneme	Pitch	MIDI	Annotation			
							Text	Phoneme	Pitch	Alignment
Opencpop	pop	5.25	1	miss “van”,“ve”,“vn”	D#3-D#5	✓	✓	✓	✓	✓
M4Singer	pop	29.77	20	no miss	G1-G5	✓	✓	✓	×	✓
OpenSongSong	SongCi	29.90	89	no miss	G1-C6	✓	✓	✓	✓	✓

Table 1: The comparison results for OpenSongSong, Opencpop, and M4Singer. The “Phenome” indicates if certain phonemes are absent from the dataset, while the “Pitch” reflects the distribution of pitches within the dataset. Phonemes or pitches that occur fewer than 20 times are deemed excluded from the dataset. The “MIDI” indicates whether the dataset includes music score information. The “Annotation” specifies the number of annotations present in the dataset. Lastly, the “Alignment” indicates whether each annotation is synchronized with the corresponding singing voice recording, allowing for the start and end times of each element to be extracted from the annotation type.

(Défossez et al. 2022). During inference, the text tokens and audio tokens are combined and fed into an auto-regressive Transformer to produce the accompaniment audio tokens. This tokens are then transformed into a continuous Mel spectrogram using the decoder from Encodec.

Experiment

Dataset

OpenSongSong is an extensive collection of Chinese SongCi music that includes recordings of singing voices about 29.9 hours along with their associated annotations and musical score details. The annotations are stored in a textgrid file, which contains text, phoneme sequences, and pitch sequences. Each annotation is synchronized with the singing voice audio, allowing for easy access to the start and end times of each element directly from the textgrid file. The musical score information is converted into MIDI format, which represents a musical composition in written form, typically detailing note pitch, duration, and tempo.

We compare the richness of Opencpop and M4Singer with OpenSongSong in Table 1. The size of OpenSongSong is comparable to that of M4Singer, and both datasets encompass all Chinese phonemes. However, OpenSongSong features a greater number of singers and a broader pitch range. Additionally, both OpenSongSong and Opencpop include pitch information in the textgrid file, eliminating the need to extract pitch from the audio again, thus facilitating its use in SVS tasks. OpenSongSong stands as the first large-scale, high-quality dataset of SongCi music, offering diverse annotations, covering all Chinese phonemes, and supporting a wide pitch distribution. It is suitable for various music-related applications, including singing voice generation, accompaniment generation, and lyrics-melody alignment.

Configuration

Both the lyric-to-rhythme and rhythm-to-melody modules share the same Transformer architecture, consisting of 4 layers of encoder blocks and 4 layers of decoder blocks. Each block features 4 attention heads with 256 linear units. In the singing voice generation module, our acoustic encoder and variance encoder are based on FastSpeech2, following the same configuration as Ren et al. (2020), which includes 4 feed-forward Transformers. The duration predictor comprises 5 layers of convolutional networks, each with a one-dimensional convolution and a kernel size of 3, maintaining an input and output size of 512. The auxiliary decoder consists of 6 ConvNeXt decoder layers, each with a kernel size of 7 and an input and output size of 512. The denoiser and F0 predictor are both based on Wavenet, featuring 20 convolutional layers with a kernel size of 3. The input and output sizes for the denoiser in each layer are 512 and 1024, respectively, while the F0 predictor has half the size. For the accompaniment generation module, we utilize 1.5B-Musicgen-melody as the initial parameters.

We utilize data from the speaker with the largest corpus from OpenSongSong, which amounts to around 3.5 hours, to train the lyrics-to-rhythm, rhythm-to-melody, and singing voice generation components of SongSong. The rhythm and melody corpora are prepared following the methodology outlined by Ju et al. (2022). All training sessions consistently use the Adam optimizer with a learning rate set at 5e-4. The training is performed on GeForce RTX-3090, with all modules trained for a maximum of 160,000 steps.

Currently, there are limited open-source systems capable of generating audio that combines both singing and accompaniment. Therefore, we select Suno and SkyMusic, two leading commercial music generation software, for comparison. These systems leverage the advanced generation capabilities of LLMs to produce high-quality music that includes both singing and accompaniment based on simple text prompts. Additionally, they allow users to input an audio reference to generate music in a similar style.

We use objective and subjective metrics for evaluation. The objective metric employed is the Frechet Audio Distance (Kilgour et al. 2018), which measures the Frechet distance between the embeddings from two audio groups, providing an evaluation of generated audio quality that aligns more closely with human perception. We utilize the VGGish and PANN (Kong et al. 2020) models to extract audio embeddings. For subjective evaluation, we evaluate the music structure, richness of instrumentation, continuity of motivation, sound quality, pronunciation accuracy, naturalness of voice, adherence to the SongCi style and accompaniment conformity. The first four metrics are evaluated from the perspective of ordinary music, while the last four metrics evaluate the generated music from the perspective of SongCi

Test	Model	Objective Metrics		Common Subjective Metrics				SongCi Subjective Metrics			
		$\mathrm{FAD_{vgg}}$ (↓)	$\mathrm{FAD_{pann}}$ (↓)	MS (↑)	ER (↑)	MC (↑)	SQ (↑)	**PA** (↑)	**VN** (↑)	**SCS** (↑)	**AC** (↑)
Zero-shot	Suno	5.74	**3.35e-4**	65.48	68.85	63.19	62.96	62.59	65.37	35.30	19.28
	SkyMusic	7.88	3.42e-4	63.96	65.48	59.11	72.67	54.81	**72.89**	25.37	45.69
	SongSong	**5.41**	3.87e-4	56.67	46.26	53.93	55.56	**75.37**	65.56	**78.44**	**65.54**
Few-shot	Suno	7.92	3.98e-4	64.44	62.30	64.41	75.15	48.85	71.04	42.19	28.90
	SkyMusic	6.31	**5.55e-5**	60.26	58.30	60.63	76.41	59.04	72.74	46.67	52.60
	SongSong	**5.41**	3.87e-4	50.52	49.48	50.04	63.74	**78.48**	**72.81**	**75.19**	**69.76**

Table 2: The comparison results for SongSong, Suno, and SkyMusic. The terms zero-shot and few-shot indicate whether Songci audio was provided as a reference for Suno and SkyMusic. For subjective metrics, MS stands for music structure, ER denotes equipment richness, MC indicates motivation continuity, PA refers to pronunciation accuracy, VN signifies voice naturality, SQ represents sound quality, SCS pertains to conformity with the SongCi style and AC is acompaniment conformity.

music.

We randomly select 50 songs that are not part of the training data from the SongCi collection used to develop OpenSongSong for our test set. After performing basic segmentation to eliminate long stretches of silence, we end up with 85 utterances totaling 1.8 hours. The singer features in this test set is male. The testing process consists of two phases. The first phase is a zero-shot test, where only text prompt and SongCi lyric are provided to Suno and SkyMusic, with the prompt specified as "Chinese classical style, Song Dynasty, Guzheng, male voice." The second phase is a few-shot test, where Suno and SkyMusic receive an additional SongCi piece as a reference. We invite 9 experts from music academies to participate in the subjective evaluation.

Main Result

The experimental results are shown in Table 2. When conducting subjective evaluations from the perspective of SongCi music, we find that Suno and SkyMusic have poor prompt acceptance ability and cannot understand Chinese classical styles well. The generated music is almost all in the style of pop music, and the musical characteristics are far from that of SongCi music. Moreover, the musical accompaniment does not use the guzheng as we have defined, but uses instruments such as guitars and keyboard instruments that are typically used in pop music. This problem is difficult to alleviate even when inputting SongCi music as a reference. Therefore, Suno and SkyMusic score much lower than SongSong on SCS and AC. SongSong, after being trained on the SongCi musical corpus in OpenSongSong, can fully grasp the pitch and rhythm of SongCi music, and generate singing voice and accompaniment that are more in line with SongCi music. In addition, due to the presence of many rare characters in Song poetry, the two large music models trained on popular music cannot clearly and accurately sing every word, and sometimes even miss words and generate random lyrics. This results in their PA being low. After few-shot, the accuracy of SkyMusic's pronunciation shows an upward trend, while that of Suno shows a downward trend. However, this issue has not significantly affected the VN of Suno and SkyMusic. With the large scale pre-training and a large number of model parameters, Suno and SkyMusic can still maintain the naturalness of the original performance to a certain extent, so VN is not lower than SongSong. SongSong has undergone strict alignment of lyrics and melody, and the training corpus already includes rare words that may appear in SongCi music, so it can not only sing strictly according to the input lyrics, but also sing each word with correct pronunciation. Therefore, it far exceeds the other two models in the PA metric and is comparable to them in VN. However, when evaluating subjectively from the perspective of ordinary music, SongSong's performance is not as good as Suno and SkyMusic. In addition to the backwardness in model size and training data capacity, there are several reasons for this phenomenon. In a SongCi poem, the sentence structure and length are relatively fixed, and the performance style is relatively monotonous. But Suno and SkyMusic sing it as a popular song, using structural optimization methods such as repetition, contrast, and modulation to generate more complex and continuous pop music, thus having higher MS and MC; The controllability of Suno and SkyMusic is poor. Although we input a prompt requiring them to only use the guzheng for accompaniment, they still use other instruments for accompaniment, resulting in a higher degree of orchestration, so their ERs are high; After few-shot, MS and ER decrease for Suno and SkyMusic, while MC remains unchanged. This observation supports our analysis that the closer the generated music is to SongCi, the more monotonous and consistent its style will be. In terms of sound quality, due to the lack of high-quality Song Dynasty music, our trained model can not generate muisc with very high sound quality. In terms of objective evaluation metrics, SongSong has a relatively high $\mathrm{FAD_{VGG}}$, but it is comparable to Suno and SkyMusic in terms of $\mathrm{FAD_{PASS}}$. The audio for comparison is SongCi audio, so objectively speaking, SongSong can also generate high-quality SongCi audio that conforms to human auditory perception.

Case Study

To showcase SongSong's capability to produce high-quality audio that aligns more closely with the style of SongCi music, we carry out a case study using a sample from a test set.

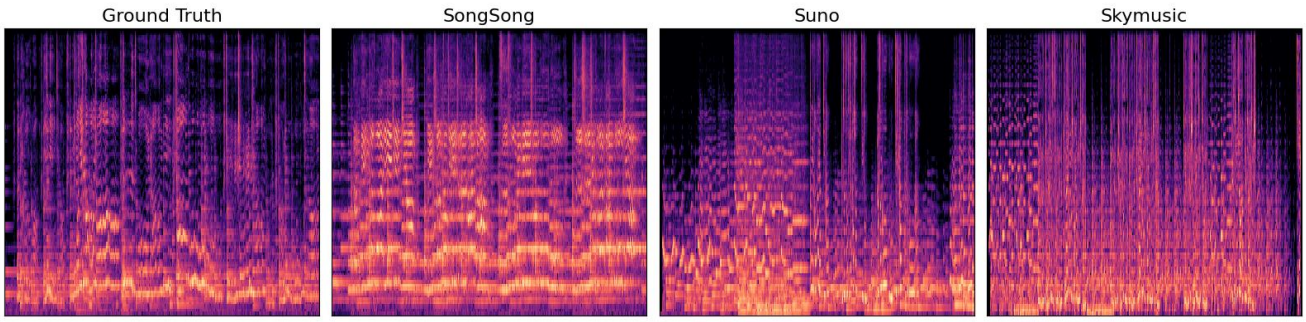

Figure 6: The mel-spectrogram comparisons among ground truth, SongSong, Suno and Skymusic.

Figure 6 displays the mel-spectrogram of the sample alongside the audio generated by SongSong, Suno, and SkyMusic based on the sample's lyrics. The original mel-spectrogram is repetitive over time due to the consistent sentence structure of the SongCi and the fixed rhythm of the SongCi music. In contrast, Suno's generated mel-spectrogram exhibits significant variation in frequency and loudness across different time intervals, suggesting a high level of randomness which shows Suno fails to capture the monotonous and repetitive nature of Song poetry. Additionally, it lacks mid-frequency and high-frequency bands. SkyMusic performs better than Suno in generating Chinese SongCi music, with its mel-spectrogram reflecting the overall style of Song poetry; however, it suffers from abrupt changes in the high-frequency range. On the other hand, the mel-spectrogram produced by SongSong shows clear temporal repetition, indicating that it has effectively learned the principles of SongCi music and can maintain a consistent rhythm for each SongCi line, performing them in accordance with the SongCi style.

Ablation Study

Model	Objective Metrics		SongCi Subjective Metrics		
	FAD_{vgg} (↓)	FAD_{pann} (↓)	PA (↑)	VN (↑)	SCS (↑)
GSV	20.30	**4.93e-4**	45.25	69.25	46.25
SongSong	**7.22**	5.05e-4	**81.25**	**70.00**	**75.50**

Table 3: The results of ablation study.

In essence, Suno adopts a structure similar to VALL-E (Wang et al. 2023), discretizing acoustic features into tokens, and then leveraging LLM to sequentially decode the acoustic tokens to generate a complete piece of music. But in fact, this method is not suitable for low-resource situations, where the melody generation module plays a key role. To prove this, we additionally introduce GPT-SoVITS (GSV), a SOTA speech synthesis model, which is based on Brown (2020) and Kim, Kong, and Son (2021), has the above architecture. It can be trained using the same training data as SongSong to obtain the ability to generate singing voice. During the evaluation, we select 20 unused SongCi audio clips and their corresponding lyrics, and ask GSV and SongSong to generate SongCi music based on the lyrics. GSV uses a piece of SongCi music for few-shot. We assess the FAD based on the original SongCi audio clips, and we also invite two music experts to evaluate the performance of the generated audio on subjective metrics of SongCi music. The experimental results are shown in Table 3. In terms of objective metrics, SongSong significantly outperforms GSV on FAD_{vgg}. For three subjective metrics of SongCi, GSV's performance is also far inferior to SongSong. During the subjective evaluation, our music experts point out that although GSV can sing SongCi with a certain rhythm, it cannot completely sing the input SongCi, and there are serious phenomena of missing words and repeated generation. Even though GSV has been trained on SongCi music data and uses few-shot in inference, it is difficult to align lyrics and melody due to limited training corpus, so it cannot strictly output based on the input SongCi. SongSong uses a rhythm-based lyric-melody alignment scheme to generate melody information first, and then uses both the lyric information and the melody information to generate singing voice, which is a high-quality audio that can fully perform the input lyrics and conform to the style of SongCi.

Conclusion

We present the first music generation model capable of performing Chinese SongCi, SongSong, and the first comprehensive dataset of ancient Chinese SongCi music, OpenSongSong, which supports the restoration of SongCi music. SongSong first predicts the melody from the input SongCi, then separately generates the singing voice and accompaniment based on that melody, and finally combines all audio elements to create the final SongCi music. OpenSongSong features 29.9 hours of SongCi music. We evaluate the performance of SongSong, Suno and SkyMusic, and the results indicate that our proposed model can produce high-quality music that embodies the SongCi style.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 62306216, No. 72074171, No. 72374161), the Natural Science Foundation of Hubei Province of China (No. 2023AFB816), the Fundamental Research Funds for the Central Universities (No. 2042023kf0133).

References

Bao, H.; Huang, S.; Wei, F.; Cui, L.; Wu, Y.; Tan, C.; Piao, S.; and Zhou, M. 2019. Neural melody composition from lyrics. In *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part I 8*, 499–511. Springer.

Brown, T. B. 2020. Language models are few-shot learners. *arXiv preprint ArXiv:2005.14165*.

Chandna, P.; Blaauw, M.; Bonada, J.; and Gómez, E. 2019. Wgansing: A multi-voice singing voice synthesizer based on the wasserstein-gan. In *2019 27th European signal processing conference (EUSIPCO)*, 1–5. IEEE.

Chen, J.; Tan, X.; Luan, J.; Qin, T.; and Liu, T.-Y. 2020. Hifisinger: Towards high-fidelity neural singing voice synthesis. *arXiv preprint arXiv:2009.01776*.

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

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

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

Hono, Y.; Hashimoto, K.; Oura, K.; Nankaku, Y.; and Tokuda, K. 2019. Singing voice synthesis based on generative adversarial networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6955–6959. IEEE.

Ju, Z.; Lu, P.; Tan, X.; Wang, R.; Zhang, C.; Wu, S.; Zhang, K.; Li, X.-Y.; Qin, T.; and Liu, T.-Y. 2022. TeleMelody: Lyric-to-Melody Generation with a Template-Based Two-Stage Method. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 5426–5437.

Kilgour, K.; Zuluaga, M.; Roblek, D.; and Sharifi, M. 2018. Fr\'echet audio distance: A metric for evaluating music enhancement algorithms. *arXiv preprint arXiv:1812.08466*.

Kim, J.; Choi, H.; Park, J.; Kim, S.; Kim, J.; and Hahn, M. 2018. Korean singing voice synthesis system based on an LSTM recurrent neural network. In *Proc. Interspeech*, 1551–1555.

Kim, J.; Kong, J.; and Son, J. 2021. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, 5530–5540. PMLR.

Kong, Q.; Cao, Y.; Iqbal, T.; Wang, Y.; Wang, W.; and Plumbley, M. D. 2020. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28: 2880–2894.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, J.; Yang, L.; Tang, M.; Chen, C.; Li, Z.; Wang, P.; and Zhao, H. 2024. The music maestro or the musically challenged, a massive music evaluation benchmark for large language models. *arXiv preprint arXiv:2406.15885*.

Liu, J.; Li, C.; Ren, Y.; Chen, F.; and Zhao, Z. 2022a. Diffsinger: Singing voice synthesis via shallow diffusion mechanism. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 11020–11028.

Liu, Z.; Mao, H.; Wu, C.-Y.; Feichtenhofer, C.; Darrell, T.; and Xie, S. 2022b. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11976–11986.

Lu, P.; Wu, J.; Luan, J.; Tan, X.; and Zhou, L. 2020. Xiaoicesing: A high-quality and integrated singing voice synthesis system. *arXiv preprint arXiv:2006.06261*.

Lv, A.; Tan, X.; Qin, T.; Liu, T.-Y.; and Yan, R. 2022. Recreation of creations: A new paradigm for lyric-to-melody generation. *arXiv preprint arXiv:2208.05697*.

Nakamura, K.; Hashimoto, K.; Oura, K.; Nankaku, Y.; and Tokuda, K. 2019. Singing voice synthesis based on convolutional neural networks. *arXiv preprint arXiv:1904.06868*.

Nakamura, K.; Takaki, S.; Hashimoto, K.; Oura, K.; Nankaku, Y.; and Tokuda, K. 2020. Fast and high-quality singing voice synthesis system based on convolutional neural networks. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7239–7243. IEEE.

Nichol, A. Q.; and Dhariwal, P. 2021. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, 8162–8171. PMLR.

Nishimura, M.; Hashimoto, K.; Oura, K.; Nankaku, Y.; and Tokuda, K. 2016. Singing Voice Synthesis Based on Deep Neural Networks. In *Interspeech*, 2478–2482.

Oord, A. v. d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; and Kavukcuoglu, K. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Rabiner, L. R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2): 257–286.

Ren, Y.; Hu, C.; Tan, X.; Qin, T.; Zhao, S.; Zhao, Z.; and Liu, T.-Y. 2020. Fastspeech 2: Fast and high-quality end-to-end text to speech. *arXiv preprint arXiv:2006.04558*.

Shan, Y.; Zhang, J.; Ren, H.; Qiu, Y.; and Zhou, J. 2023. LingGe: An Automatic Ancient Chinese Poem-to-Song Generation System. In *IJCAI*, 7171–7174.

Sheng, Z.; Song, K.; Tan, X.; Ren, Y.; Ye, W.; Zhang, S.; and Qin, T. 2021. Songmass: Automatic song writing with pre-training and alignment constraint. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 13798–13805.

Tian, J.; Li, Z.; Li, J.; and Wang, P. 2024. N-gram Unsupervised Compoundation and Feature Injection for Better Symbolic Music Understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 15364–15372.

Vaswani, A. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Wang, C.; Chen, S.; Wu, Y.; Zhang, Z.; Zhou, L.; Liu, S.; Chen, Z.; Liu, Y.; Wang, H.; Li, J.; et al. 2023. Neural codec language models are zero-shot text to speech synthesizers. *arXiv preprint arXiv:2301.02111*.

Wang, Y.; Wang, X.; Zhu, P.; Wu, J.; Li, H.; Xue, H.; Zhang, Y.; Xie, L.; and Bi, M. 2022. Opencpop: A high-quality open source chinese popular song corpus for singing voice synthesis. *arXiv preprint arXiv:2201.07429*.

Yamamoto, R.; Song, E.; and Kim, J.-M. 2020. Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6199–6203. IEEE.

Yu, Y.; Srivastava, A.; and Canales, S. 2021. Conditional LSTM-GAN for melody generation from lyrics. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 17(1): 1–20.

Zhang, C.; Chang, L.; Wu, S.; Tan, X.; Qin, T.; Liu, T.-Y.; and Zhang, K. 2022a. Relyme: improving lyric-to-melody generation by incorporating lyric-melody relationships. In *Proceedings of the 30th ACM International Conference on Multimedia*, 1047–1056.

Zhang, L.; Li, R.; Wang, S.; Deng, L.; Liu, J.; Ren, Y.; He, J.; Huang, R.; Zhu, J.; Chen, X.; et al. 2022b. M4singer: A multi-style, multi-singer and musical score provided mandarin singing corpus. *Advances in Neural Information Processing Systems*, 35: 6914–6926.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.