Université de Montréal

Automatic Symbolic Melody Generation from Lyrics

par

Yifan Xie

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

August 25, 2023

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Automatic Symbolic Melody Generation from Lyrics

présenté par

Yifan Xie

a été évalué par un jury composé des personnes suivantes :

Philippe Langlais
(président-rapporteur)

Jian-Yun Nie
(directeur de recherche)

Bang Liu
(membre du jury)

Résumé

La génération de musique est une tâche populaire dans le domaine de l'intelligence artificielle musicale, visant à générer automatiquement de la musique. La génération musicale comprend la génération de musique symbolique et acoustique. La première se concentre sur le niveau de la partition, tandis que la seconde met l'accent sur le niveau du signal audio. Ce mémoire se concentre sur une tâche de génération musicale symbolique : générer des mélodies symboliques à partir de paroles et tenter de résoudre plusieurs problèmes existants dans ce domaine.

Premièrement, nous abordons le problème de génération de la mélodie à partir de la parole pour la musique non populaire, un problème assez peu étudié. Nous étudions non seulement la génération de la musique populaire à partir de la parole en anglais, mais aussi et surtout de la musique chinoise traditionnelle avec de la poésie classique. La première a fait l'objet de nombreuses recherches, tandis que la dernière a rarement été explorée.

Deuxièmement, pour atténuer le défi de la modélisation insuffisante de la relation entre les paroles et la mélodie dans la musique non populaire, nous utilisons des réseaux neuronaux profonds pour apprendre à partir d'un ensemble de données appariées plus grand pour générer des mélodies à partir de la poésie chinoise classique. Cette approche renforce la capacité du modèle à comprendre la relation entre la poésie chinoise classique et ses mélodies associées. Une autre motivation derrière cette démarche provient du contexte historique : de nombreux poèmes chinois classiques pouvaient être chantés dans l'Antiquité, mais de nombreuses mélodies associées ont été perdues, ne laissant que la poésie elle-même. En supposant que les mélodies perdues partagent des éléments similaires, tels que les styles et les genres, avec les mélodies préservées, ce mémoire utilise des réseaux neuronaux profonds pour modéliser les mélodies restantes et leurs poèmes correspondants, ce qui peut aider à restaurer ces mélodies perdues.

Troisièmement, la recherche précédente intègre des règles musicales humaines pour améliorer les performances, ce qui a des limitations en matière de généralisation et d'adaptabilité. Nous employons des méthodes permettant au modèle de coder de manière autonome des informations théoriques sur la musique pour la génération de mélodies. Plus précisément, des plongements de parties du discours et des plongements de tons sont intégrés

au modèle, améliorant la capture des relations entre les frontières prosodiques dans les paroles (applicables à la fois aux paroles anglaises et chinoises) et la mélodie, ainsi qu'entre le ton des caractères chinois et la hauteur de la mélodie, sans règles conçues manuellement.

Quatrièmement, pour aborder le problème du manque de caractéristiques stylistiques des mélodies générées, nous intégrons des contraintes de style dans la phase d'inférence. Cet ajustement permet au modèle de saisir dans une certaine mesure les caractéristiques stylistiques globales de la musique.

Après avoir mis en œuvre ces adaptations, des évaluations objectives et subjectives sont menées. Les études objectives d'ablation confirment que chaque adaptation contribue à améliorer l'ajustement du modèle aux données. Les évaluations subjectives corroborent que notre modèle peut générer des mélodies de haute qualité semblables à de la vraie musique.

Mots-clés: Génération de musique, Génération automatique de mélodie, Paroles, Transformateur

Abstract

Music generation is a popular task in the domain of music artificial intelligence, aiming at generating music automatically. Music generation includes both symbolic and acoustic music generation. The former focuses on the score level, while the latter emphasizes the audio signal level. This thesis focuses on one task of symbolic music generation: generating symbolic melodies from lyrics and attempting to solve several pre-existing issues in this field.

Firstly, we address the problem of melody generation from lyrics for non-popular music, which has not been widely studied in the literature, in addition to the generation of popular music. We study the following two music types: popular music with English lyrics and traditional Chinese music with classical Chinese poetry. The former has been extensively researched, while the latter has seldom been explored.

Secondly, to mitigate the challenge of insufficient modeling of the relationship between lyrics and melody in non-popular music, we utilize deep neural networks to learn from a larger paired dataset for generating melodies from classical Chinese poetry. This approach enhances the model's ability to understand the relationship between classical Chinese poetry and its associated melodies. Another motivation behind this endeavor stems from historical context: many classical Chinese poems could be sung in ancient times, but many associated melodies have been lost, leaving only the poetry itself. Given the assumption that the lost melodies share similar elements, such as styles and genres, with the preserved melodies, this thesis employs deep neural networks to model the remaining melodies and their corresponding poems, which may assist in restoring these lost melodies.

Thirdly, prior research integrates human music rules to enhance performance, which has limitations in generalization and adaptability. To tackle this issue, we employ methods allowing the model to autonomously encode music theory information for melody generation. Specifically, part-of-speech embeddings and tone embeddings are incorporated into the model, improving the capture of relationships between prosodic boundaries in lyrics (applicable to both English and Chinese lyrics) and melody, as well as between the tone of Chinese characters and the pitch of the melody, without manually designed rules.

Fourthly, to address the problem of generated melodies lacking stylistic features, we incorporate style constraints into the inference phase. This adjustment enables the model to grasp the global style features of music to some extent.

After implementing these adaptations, both objective and subjective evaluations are conducted. Objective ablation studies confirm that each adaptation contributes to improving the model's fit to the data. Subjective evaluations corroborate that our model can generate high-quality melodies akin to real music.

Keywords: Music generation, Automatic melody generation, Lyrics, Transformer

Contents

Résumé 5

Abstract 7

List of tables 13

List of figures 15

List of Abbreviations 17

Acknowledgements 19

Chapter 1. Introduction 21

1.1. Introduction to AI + Music 21

1.1.1. Introduction of Music Representation 21

1.1.2. Introduction of AI + Music Tasks 24

1.2. Melody Generation from Lyrics Task 25

1.3. Problems in Melody Generation from Lyrics 25

1.4. Outlook of Our Solutions 26

1.5. Organization 28

Chapter 2. Background and Related Work 29

2.1. Background on Music Theory 29

2.1.1. Staff Score and Related Notations 29

2.1.2. Numbered Musical Score and Related Notations 31

2.1.3. Gongchepu Score and Related Notations 33

2.2. Background on Symbolic Music Generation 34

2.2.1. Monophonic Symbolic Music Generation 34

2.2.2. Polyphonic Symbolic Music Generation 34

2.3. Related Work on Melody Generation from Lyrics 35

2.3.1. RNN Revisted and RNN-Based Approaches 35
2.3.2. Transformer Revisted and Transformer-Based Approaches 40
2.3.3. Music Genres Explored in Existing Melody Generation from Lyrics Works . 43

Chapter 3. Dataset, Data Preprocessing, and Data Pepresentation 45

3.1. Dataset 45
3.1.1. Dataset with English Lyrics 45
3.1.2. Dataset with Chinese Lyrics 46

3.2. Chinese lyrics Data Preprocessing 48
3.2.1. Pitch Preprocessing 48
3.2.2. Duration Preprocessing 49
3.2.3. A Entire Transform Example from Gongchepu to Staff Score 51

3.3. Data Representation 51

Chapter 4. Melody Generation System and Training Method 53

4.1. Basic Framework and Traning Method 53
4.1.1. Basic Model Framework 53
4.1.2. Training Methods for the Chinese lyrics dataset 55
4.1.3. Training Methods for the English lyrics dataset 55

4.2. Incorporating POS and Tone Embeddings 55
4.2.1. Motivations for Incorporating POS Embeddings 56
4.2.2. Motivations for Incorporating Tone Embeddings 56
4.2.3. Methods to incorporate Tone and POS Embeddings 57

4.3. Re-ranking Generation Candidates Based on Style during the Inference Stage 59
4.3.1. Motivations to Utilize Gongdiao-Style 59
4.3.2. The Method to Add Gongdiao-Style Constrain 59

Chapter 5. Implementation and Experiments 61

5.1. Implementation Details 61

5.2. Automatic Global Performance Evaluation 62

5.3. Ablation Studies 62
5.3.1. Effect Analysis of Incorporating POS Embeddings 63
5.3.2. Effect Analysis of Incorporating Tone Embeddings 64

5.3.3. Effect of Re-ranking Generation Candidates Based on Style 64
5.4. Human Evaluation of the Generated Songs 66
5.5. Melody Generation Examples 68
5.5.1. Melody Generation Examples from English Lyrics 68
5.5.2. Melody Generation Examples from Chinese Lyrics 69
Chapter 6. Conclusions and Limitations 71
6.1. Conclusions 71
6.2. Limitations and Future Work 72
References 73

List of tables

2.1 Pitch transformation between staff score and numbered musical score if 1 = C5 . 32

2.2 Pitch transformation between staff score and numbered musical score if 1 = G5 . 33

3.1 Example of Figure 3.3's song after digitization 48

3.2 Example of Figure 3.5's song after digitization 48

3.3 Pitch transformation when seeing 上 as the standard (Part 1). This table shows part of the pitches used in the dataset..... 49

3.4 Pitch transformation when seeing 上 as the standard (Part 2). This table shows the remaining pitches used in the dataset..... 49

3.5 Rhythm symbols in the used dataset. Notice the difference between □ and □ 50

5.1 The hyper-parameters of the model 62

5.2 Automatic global performance evaluation in Chinese lyrics dataset 62

5.3 The best validation NLL between baseline and incorporating tone or POS embedding in English lyrics dataset 63

5.4 The best validation NLL between baseline and incorporating tone or POS embedding in Chinese lyrics dataset 63

5.5 The best validation NLL between baseline and incorporating tone or POS embedding in Chinese lyrics dataset 64

5.6 The Pitch Distribution Similarity scores between baseline and adding Gongdiao-Style constraint 64

5.7 Results of the subjective evaluation. "Win" means the first item in this pair wins, and "Loss" means the second item in this pair wins. The numbers below 'wins', 'ties', and 'losses' represent the total count of participants choosing each option. * denotes there are significant differences (p value < 0.05) between compared pairs under independent two-sample t-test. 67

List of figures

1.1 An example of a music score excerpt 22
1.2 The waveform of an audio signal from a music excerpt 23
1.3 Spectrogram of the audio signal from a music excerpt 23
1.4 An example includes the melody and the lyrics for popular English music 25

2.1 A music excerpt in staff score form 29
2.2 The explanation of Figure 2.1 30
2.3 An example demonstrating the connection between the scientific pitch notation and the staff notation 31
2.4 An introduction to duration. The figure is sourced from [**1**]. 31
2.5 An example of a music excerpt in numbered musical notation 32
2.6 Note durations in numbered musical scores 33
2.7 A music excerpt illustrating a monophony melody. This excerpt comes from *Echigo-Jishi*, which is arranged by Y. Nagai and K. Kobatake. 34
2.8 An music excerpt illustrating polyphony melody from the same timbre. This excerpt comes from *An Chloe, K. 524*, composed by W. A. Mozart. 35
2.9 A music excerpt illustrating polyphony melody from the same timbre. This excerpt comes from *An die ferne Geliebte, Op. 98*, composed by L. van Beethoven. 35
2.10 The vanilla RNN architecture, referred from [**51**] 36
2.11 The Long Short-Term Memory network architecture, referred from [**51**] 37
2.12 The Gated Recurrent Unit architecture, referred from [**51**] 38
2.13 The architecture of the Transformer model. Referred from [**57**]. 41

3.1 A syllable-note alignment example of an English song excerpt from [**2**]. This figure is taken from [**64**]. 45
3.2 A word-note alignment example of an English song excerpt from [**2**]. This figure is pasted from [**53**]. 46

3.3 A Gongchepu example from `Jiugong Dacheng Nanbei Ci Gongpu` 47
3.4 The explanation of dividing beats 50
3.5 An example from Gongchepu to staff 51
3.6 Data Representation Example of Chinese lyrics song 52

4.1 The basic model framework. This figure is pasted from [**53**] 53
4.2 The unsupervised pretraining in the SongMASS model. This figure is pasted from [**53**] 54
4.3 An example showing how pitches in melody related to tones in lyrics from [**25**] .. 56
4.4 An example showing the lyrics with the same structures tend to have similar pitch flows 57
4.5 Incoporating Tone Embedding and Part-of-Speech Embedding into lyrics encoder and lyrics decoder 58

5.1 The confusion matrix of the Gongdiao-style classifier 65
5.2 A melody generation example from English lyrics 68
5.3 A melody generation example from English lyrics 69
5.4 The original song of Figure 5.2 69
5.5 One song from this dataset. The English translation of the lyrics in this song is “Cold, oh so cold; melancholy harbors deep within, hurting the heart; sweet dreams are hard to fulfill. The bright moon pierces through the window; casting its light upon me, solely keeping company with solitude." 69
5.6 One song generated by our model. The English translation of the lyrics in this song is “Morning rain in Weicheng dampens the light dust. The guest house is lush with the color of new willows. I urge you to finish yet another cup of wine. As west of Yangguan, there will be no old friends." 69
5.7 Melody generation example from Chinese lyrics based on our model. The English translation of the lyrics in this song is “The zither and the crane, motion, and stillness take their forms; Such is the lofty and solitary nature of a child. See, in the deep night, the celestial bird responds to the intoxicating sound; Accompanying me, under the full moon, the river, the sky, and the earth are all green." 70
5.8 A comparison of the generated melodies between SongMASS and our model 70

List of Abbreviations

AI	*Artificial Intelligence*
CNN	*Convolutional Neural Network*
RNN	*Recurrent Neural Network*
BPTT	*Back-Propagation through Time*
LSTM	*Long Short-Term Memory Network*
GRU	*Gated Recurrent Unit*
GAN	*Generative Adversarial Network*
FFN	*Feed-Forward Network*
POS	*Part-of-Specch*
NLL	*Negative Log-Likelihood*
PD	*Pitch Distribution Similarity*

DD *Duration Distribution Similarity*

Acknowledgements

I am deeply grateful to my supervisor, Jian-Yun Nie, for giving me the opportunity to become his student. I truly appreciate and cherish this experience. I have always been able to get valuable suggestions at every stage of the research process. I also thank him for his meticulous guidance and revision to improve the quality of my written work.

Furthermore, I would like to express my gratitude to all my friends who have accompanied me on this journey in Montréal over the past two years. These times have been both happy and difficult, and I am grateful for their companionship during these periods.

Finally, I would like to thank my parents who have supported me all along. I am very grateful for the strong love I have received from my parents.

Chapter 1

Introduction

Artificial intelligence (AI) has been used to help in many human tasks, including natural language processing and music processing. This thesis focuses on the utilization of AI for music. There is a wide range of tasks that AI can address in music, ranging from music classification, music synthesis to music generation.

This thesis focuses on music generation from lyrics by taking into account the music style and linguistic features. Before presenting this specific task, in this chapter, we provide a comprehensive introduction to these tasks, which we will call AI + Music. This introduction serves to position this thesis within the broader AI field and identify the specific task addressed by this thesis. After this general introduction, we delve into the challenges that previous research has encountered in relation to melody generation from lyrics, which is the focus of this thesis. Subsequently, we offer an outlook on our proposed solutions. Finally, we outline the organization of the subsequent chapters.

1.1. Introduction to AI + Music

In this section, we provide an overview of the AI + Music field. We begin by elucidating three distinct music representation methods, which serve to establish the relationship between music and the domains of vision, language, and audio. Following this, we introduce some of the prevalent tasks in the AI + Music domain. These discussions clarify the position of this thesis within the broader AI field and specify the particular task it will address.

1.1.1. Introduction of Music Representation

Artificial Intelligence techniques are nowadays widely used in vision, language, and audio. Although AI + Music is a relatively narrow field compared to natural language and vision, it is an area where AI techniques can greatly contribute to augment our capability to process music, both in understanding and generation.

Music is naturally multi-modal data that combines audio, language, and vision. Music can be represented in different modalities.

(1) **Music language representation:** Music can be represented in the form of music language. For instance, Figure 1.1 shows a melody excerpt represented in a symbolic musical score.

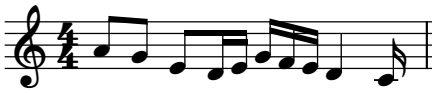


Fig. 1.1. An example of a music score excerpt

A symbolic musical score can be viewed as a unique form of language. Before we delve into specifics, it is worth noting that music, like any language, is composed of different elements. Two fundamental elements in music are 'pitch' and 'duration', representing respectively the frequency of a note and how long it lasts. In this particular example, if we were to represent a melody as a sequence of notes, with each note characterized by its pitch and duration, this melody could be represented as follows:

$$[\text{Note}_{1,\text{pitch}}, \text{Note}_{1,\text{duration}}, \text{Note}_{2,\text{pitch}}, \text{Note}_{2,\text{duration}}, ...].$$

in which the pitch and duration of each note are encoded. One specific representation of a note is to use the scientific pitch notation[1] for its pitch and a number representing the quarter length for its duration (e.g. The number 1 denotes 1 quarter note, and the number 2 denotes 1 half note ...). The representation of the melody excerpt could be:

$$[\text{D2}, 1, \text{D1}, 1, \text{C6}, 1, \text{C5}, 0.5, \text{C6}, 0.5, \text{D1}, 0.5, \text{C7}, 0.5, \text{C6}, 0.5, \text{C5}, 1, \text{C4}, 0.5]$$

This forms a symbolic sequence that can be compared to a sequence of natural language. One could imagine that some techniques developed for natural language processing (NLP) could be borrowed for music processing. Furthermore, if a musical piece includes lyrics, these lyrics are naturally a part of an extended language.

(2) **Audio-form representation:** Another usual representation of music is with its audio signal. For instance, Figure 1.2 displays a waveform generated from a music excerpt's audio signal. The original audio signal is a continuous analog signal. After sampling, the original continuous signals are transformed into discrete signals. Here, the x-axis denotes the sample number. A typical sampling rate is 44,100 Hz, indicating that the sampling process occurs 44,100 times per second. The y-axis represents the amplitude, which determines the loudness of the sound. The greater the amplitude, the louder the sound we hear.

(3) **Visual-form representation:** Music can also be represented in a visual form. For example, a spectrogram, a visual depiction of an audio signal, represents the energy

[1] `https://www.musicandtheory.com/an-easy-guide-to-scientific-pitch-notation/`

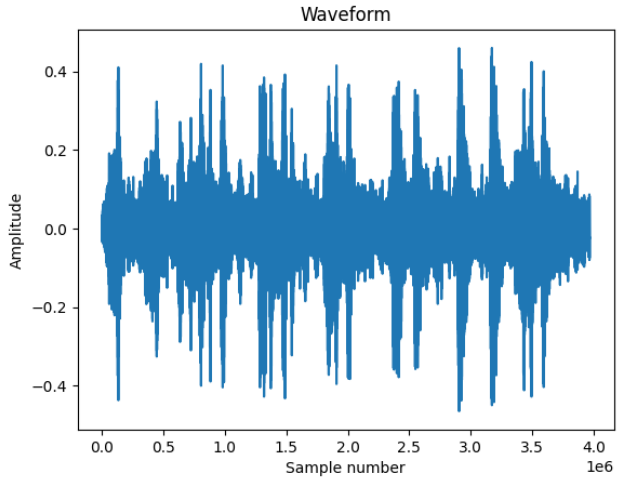


Fig. 1.2. The waveform of an audio signal from a music excerpt

distribution across various frequencies. An example of such a spectrogram is shown in Figure 1.3. This type of visual-form representation can be processed using techniques borrowed from the field of computer vision. For instance, Convolutional Neural Networks (CNNs) have been widely employed in some range of AI + Music tasks, including but not limited to instrument audio recognition [**54**], audio generation [**58, 49**], etc. All of these tasks consider spectrograms as visual representations.

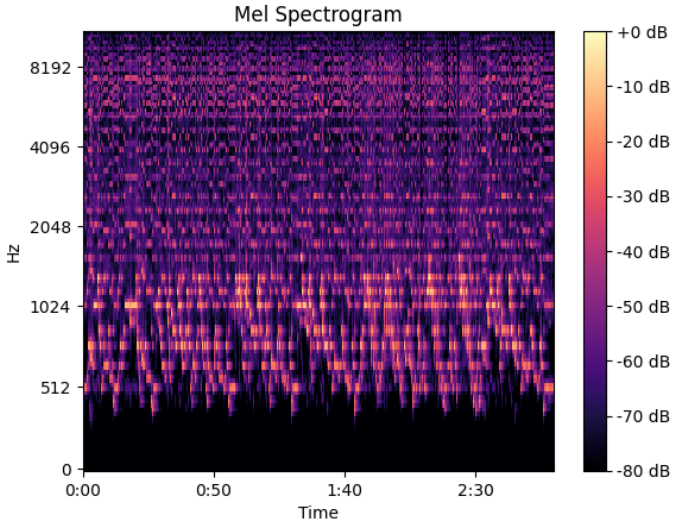


Fig. 1.3. Spectrogram of the audio signal from a music excerpt

In addition to the above problems of music representation, there is a wide range of applications based on such music representations, as well as the associated lyrics in natural language. This thesis focuses on one such task - generating melodies from lyrics. We limit our scope to the language-form representation of music, treating both symbolic melodies and

lyrics as different forms of language. To provide a more complete picture, in the following section, we briefly describe different tasks on music.

1.1.2. Introduction of AI + Music Tasks

The field of AI + Music encompasses a wide array of tasks, some of the most prevalent ones are as follows:

- **Music Generation:** Arguably the most popular task within the AI + Music domain, music generation can be divided into symbolic and acoustic music generation. The former focuses on generating music on the score level, such as the sequence of pitches and duration, while the latter focuses on generating music on the audio signal level. This thesis is focused on symbolic music generation. Symbolic music generation tasks, generally speaking, fall into two main categories: monophonic music generation [**16, 66, 35, 8**] and polyphonic music generation [**33, 21, 20, 14**]. Monophonic music involves playing a single melody line at a time, while polyphonic music allows for the concurrent play of multiple musical lines. These lines could constitute several melodies, a single melody with harmonies, or multiple melodies supported by harmonies.
- **Music Synthesis:** Music synthesis usually refers to the synthesis of audio signals from symbolic scores, including instrumental audio signals [**60, 58, 29, 22**] or singing audio [**42, 26, 50, 41**]. Music synthesis is different from acoustic music generation: Acoustic music generation focuses on creating new music pieces, while music synthesis focuses on rendering existing symbolic scores in audio. Typically, the inputs for the music synthesis tasks are musical scores, which contain melody and lyrics for singing synthesis, or melody and harmony for instrumental audio synthesis. The output is audio signals.
- **Automatic Music Transcription:** Automatic music transcription [**11**] entails "transcribing music audio into music notation" (from [**11**]). Here, music notation often refers to MIDI or musical scores. Given the complexity of certain instruments and dataset limitations, the current research on automatic music transcription is frequently conducted on instruments such as the piano [**30, 37**], guitar [**7, 17**], and human voice [**46, 19**].
- **Music Classification:** Music classification is another widely practiced task within the AI + Music domain. It can be performed at both the symbolic and audio levels. Standard music classification tasks encompass instrument recognition [**27**], music genre classification [**48**], and music emotion recognition [**28**], etc.

The tasks mentioned above are just some of the many explored in the field of AI + Music. Other tasks include Music Source Separation [**56, 44**] - like separating human singing

from accompaniment, and Optical Music Recognition [**9, 15**] - which involves converting images into digital musical scores, etc. This thesis specifically investigates the task of melody generation from lyrics, which falls under the monophonic music generation category. We describe this task in more detail in the following section.

1.2. Melody Generation from Lyrics Task

As previously stated, the task of generating melody from lyrics is part of the monophonic music generation domain, where both melody and lyrics are handled in symbolic format in this thesis. In recent years, there have been significant advances in this area. Early research [**8, 39, 64**] typically employed RNN-based methods. However, later studies [**53, 35, 66, 43, 24, 47**] shifted towards Transformer architectures. Nowadays, many researchers prefer Transformer-based models to generate melodies from lyrics due to the Transformer's enhanced modeling performance with music data. Besides, most existing studies on this topic focus on popular music, likely because of its broad appeal and available training data. For instance, Figure 1.4 displays a music excerpt with melody and lyrics from popular English music. The goal of this task is to generate the shown melody using the provided lyrics.

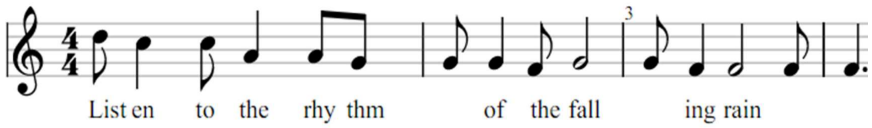


Fig. 1.4. An example includes the melody and the lyrics for popular English music

1.3. Problems in Melody Generation from Lyrics

There are still several problems in the melody generation from the lyrics task:

- **Limited coverage of non-popular music, and inadequate learning of the relationship between lyrics and melody in these contexts.**
  Existing research on melody generation from lyrics primarily centers on popular music. However, this focus tends to overlook a vast range of non-popular music, which also forms an integral part of our historical and global musical heritage. The distinct characteristic of less popular music is the limited training data, which makes it difficult to solely rely on machine learning techniques for music generation. There is great value in broadening the scope of the investigation to include less-studied types of music. While there has been some research into non-popular music—for instance, [**40, 47**] explored generating melodies from classical Chinese poetry—such studies have either been limited to employing traditional machine learning models such as the Hidden Markov Model [**10**] and the Conditional Random Field [**38**] to model a

small paired dataset or employing deep learning methods to model unpaired data, rather than directly modeling from paired data, due to the scarcity of large paired datasets. Both approaches have proven inadequate for effectively capturing the relationship between classical Chinese poetry and corresponding melodies. Classical Chinese poetry refers to poems composed in classical Chinese from the pre-Qin period (prior to 221 BC) to the end of the Qing Dynasty in 1911. In ancient times, many classical Chinese poems were designed to be sung, with verses paired with melodies. However, this type of music falls into the non-popular category and has received limited research attention. Existing studies have struggled to accurately grasp the connection between classical Chinese poetry and its accompanying melodies.

- **Lack of generalizability and adaptability with manual rules for connecting lyrics and melody.**
  Prior research has attempted to integrate music theory rules about the relationship between lyrics and melody to enhance the performance of melody generation from lyrics. For instance, [**66**] devised manual rules to leverage elements such as tone, rhythm, and structure. For instance, in terms of tone, a rule is defined as follows: when a lyric corresponds to several notes, if the tone change of this lyric aligns with the pitch changes of the corresponding notes, then such notes are more likely to be generated. However, the utilization of such rules poses considerable limitations due to their lack of generalizability. When faced with diverse musical styles, it becomes necessary to formulate new, customized rules, a process that requires expertise and can be labor-intensive. Furthermore, even within the same genre, manually crafted rules may not account for all possible scenarios, leading to potential violations under different conditions.
- **Lack of style features in generated melodies.**
  Previous studies have employed various methods to improve the alignment of different features between melody and lyrics. However, to our knowledge, there is no study to date trying to leverage style features for this task. Style represents a global attribute of music, encapsulating the overall traits of a musical piece. The music style is an important feature that influences the generation of melodies. The lack of consideration of this factor in the existing studies is likely due to the absence of precise style labels. However, the dataset used in this thesis, which is also the first time used for a machine learning task, contains such style labels.

1.4. Outlook of Our Solutions

In this thesis, we employ deep learning models to generate melodies from both popular English songs and traditional Chinese songs (as exemplified in Figure **??**). Specifically, to

address the challenges previously mentioned regarding melody generation from lyrics, this thesis tackles the following tasks:

- To **address the issue of non-popular music being rarely considered**, we generate melodies from lyrics in two different kinds of music: one is popular music with English lyrics, and the other is traditional Chinese music with classical Chinese poetry. The former has been extensively researched, while the latter has seldom been explored.
- Furthermore, to **alleviate the problem of inadequate modeling of the connection between lyrics and melody in non-popular music**, we employ deep neural networks to learn from a much larger paired dataset for generating melody from classical Chinese poetry. This enhances the model's ability to understand the relationship between classical Chinese poetry and its associated melodies. Another motivation behind this endeavor stems from historical context: many classical Chinese poems could be sung in ancient times, but many associated melodies have been lost, leaving only the poetry itself. Given the assumption that the lost melodies share similar elements, such as styles and genres, with the preserved melodies, this thesis employs deep neural networks to model the remaining melodies and their corresponding poems, which may assist in restoring these lost melodies.
- To **address the problem of generalization and adaptability of manual rules**, we employ methods allowing the model to automatically encode music theory information for melody generation. Specifically, we try to make a stronger connection between linguistic characteristics and melody. Part-of-speech (POS) embeddings and tone embeddings are incorporated into the model, improving the capture of relationships between prosodic boundaries in lyrics (applicable to both English and Chinese lyrics) and melody, as well as between the tone of Chinese language and the pitch of the melody, without manually designed rules.
- To **tackle the issue where generated melodies do not encompass stylistic features**, we integrate style constraints into the inference stage. Specifically, we initially generate some melody candidates for each lyrics sequence. From these candidates, we enhance the final generation probability of those classified as sharing the same style label with the ground truth, as determined by a style classifier. This adaptation allows the model to grasp the global style features of music to some extent.

The above ideas are implemented in a melody generation system. The generated songs (melody together with lyrics) are rendered using a music synthesis system. We conduct both objective and subjective evaluations. Objective ablation studies demonstrate that all of these adaptations contribute to improving the model's fit to the data. The results of the

subjective evaluation reveal that our model can generate high-quality melodies akin to real music.

1.5. Organization

The structure of the subsequent chapters is organized as follows:

In Chapter 2, we delve into the background and related work. Specifically, we present an overview of music theory, elaborate on tasks relevant to symbolic music generation, and conduct a comprehensive review of prior work in the domain of melody generation from lyrics.

In Chapter 3, we elucidate the datasets utilized in this thesis, the data preprocessing methods adopted, and the method of data representation. We explore two different types of music: one category encompasses popular music accompanied by English lyrics, while the other pertains to traditional Chinese music associated with classical Chinese poetry.

In Chapter 4, we present the architecture of our melody generation system and our model training methodology. We first outline the basic framework of the melody generation system, then the integration of Part-of-Speech and tone embeddings. Subsequently, we delve into the method of incorporating music style features into the system.

In Chapter 5, we introduce the experiments conducted. Beginning with an overview of the implementation, we progress to the results of objective ablation experiments. Following this, we present the outcomes of the subjective evaluation carried out by human listeners, concluding with a showcase of selected examples of melody generation.

Finally, in Chapter 6, we provide a conclusion to this study, identify its limitations, and suggest potential directions for future research. Overall, through this study, we show that melody generation from lyrics can be improved by incorporating linguistic features such as part-of-speech and tones in Chinese, as well as music style information. Such features could be applied more generally in other music generation tasks.

Chapter 2

Background and Related Work

In this chapter, we delve into the background and related work. We first present an overview of music theory, introducing three distinct forms of musical score that are used in this thesis: staff score, numbered musical score, and Gongche score in traditional Chinese music. Then, we discuss the background of symbolic music generation, with a particular focus on both monophonic and polyphonic music generation. Lastly, we conduct a review of previous research on melody generation from lyrics. This survey is based on two classification criteria: one is the models employed, including Recurrent Neural Network (RNN) and Transformer, and the other is the musical genres explored in prior studies.

2.1. Background on Music Theory

"A musical score serves as a written depiction of a musical composition, typically in a standard form of notation." ([**3**]). Essentially, musical scores guide performers in executing a musical piece. In this section, we present three distinct types of musical scores, each accompanied by their respective notations. The three scores include the staff score, numbered musical score, and Gongche score.

2.1.1. Staff Score and Related Notations

Figure 2.1 showcases a musical excerpt in the form of staff notation, while Figure 2.2 elucidates some fundamental components of the staff notation depicted in Figure 2.1. The essential elements are enclosed within circles, as illustrated in Figure 2.1and explained below.

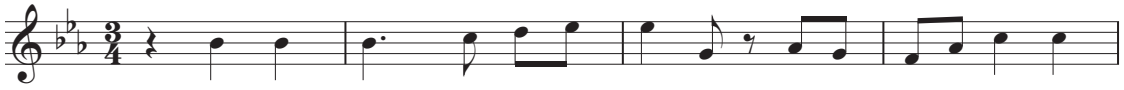


Fig. 2.1. A music excerpt in staff score form

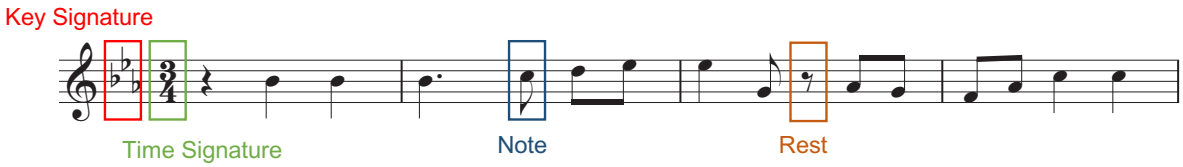


Fig. 2.2. The explanation of Figure 2.1

- **Note:** A note symbolizes a musical sound and is fundamentally characterized by two attributes: pitch and duration. The pitch represents the frequency of the sound depicted by the note, while duration indicates the length of the note. Before introducing the pitch notation in the staff score, we first introduce the scientific pitch notation. For all scores referenced in this thesis, their pitch notations can be built a direct connection with the scientific pitch notation. For instance, the range of pitch on a piano generally extends from A0 to C8 when represented in scientific pitch notation (A0, A#0, B0, C1, C#1, D1, D#1, E1, F1, F#1, G1, G#1, A1, A#1, B1, ..., C8), where A4 corresponds to 440 Hz. Every progression from A_n to A_{n+1} encompasses 12 pitches, and the frequency difference between two successive notes remains constant. This concept, known as the *12-tone equal temperament*, is a standard in Western music. Frequencies for pitches other than A4 can be computed using the following formula:

$$f = f_0 \times 2^{n/12}$$

In this formula, f represents the frequency of the target note. If we take A4 as the reference note f_0, then n is the number of semitones between the reference and the target note. For example, the frequency of B4, which is two semitones higher than A4, is calculated as follows:

$$f = f_0 \times 2^{n/12} = 440 \times 2^{2/12} \approx 493.883$$

Similarly, the frequency of F#3, which is three semitones lower than A4, is:

$$f = f_0 \times 2^{n/12} = 440 \times 2^{-3/12} \approx 369.994$$

Additionally, Figure 2.3 demonstrates how some notes expressed in scientific pitch notation are represented on a staff score.

Having introduced the concept of pitch, we now turn to duration. Figure 2.4 explains the representation of duration. If we consider a quarter note as the reference note, defining its duration as 1 quarter length, then the durations of a whole note, half note, eighth note, sixteenth note, and thirty-second note correspond to 4, 2, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$ quarter lengths, respectively. Moreover, a rest note is considered a special type of note, representing silence. Although it lacks pitch, it has a corresponding duration, which is also depicted in Figure 2.4.

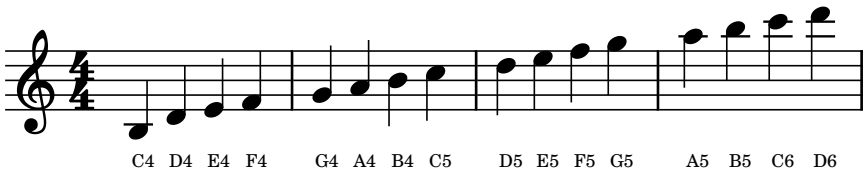


Fig. 2.3. An example demonstrating the connection between the scientific pitch notation and the staff notation

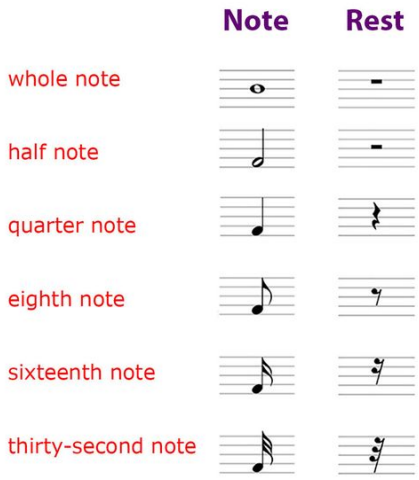


Fig. 2.4. An introduction to duration. The figure is sourced from [**1**].

- **Key Signature:** The key signature typically consists of a series of sharp (#) or flat (b) symbols located at the beginning of a musical piece. For instance, if a flat symbol appears at the A4 position in the key signature, all the A pitches in this piece are generally performed as *Ab*.
- **Time Signature:** Following the key signature in a staff score, a pair of numbers, known as the time signature, is typically presented. The bottom number designates the duration of each beat, whereas the top number indicates the number of beats within each measure (the interval between every two bar lines). For instance, the time signature (3, 4) in a staff score implies that in this musical section, the duration of each beat equals a quarter note length, and each measure comprises three beats. Similarly, (4, 8) signifies that the duration of each beat equals the length of an eighth note, and each measure contains four beats.

2.1.2. Numbered Musical Score and Related Notations

In this section, we present a notation of numbered musical scores, also known as Jianpu. The numbered musical score is a simplified form of musical score predominantly utilized

in some Asian countries, particularly China. Figure 2.5 illustrates a music excerpt represented in the numbered musical score. The subsequent descriptions elucidate how notes, key signatures, and time signatures are represented in numbered musical notation.

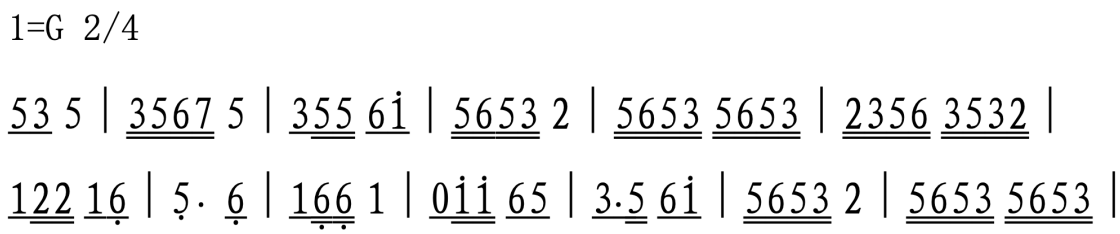


Fig. 2.5. An example of a music excerpt in numbered musical notation

- **Pitch and Key Signature in Notes:** First, we introduce the representation of pitch in notes. In numbered musical scores, pitches are primarily indicated by numbers. These numerical representations correspond to relative pitch, as opposed to the absolute pitch used in staff scores. However, we can achieve the conversion between pitch representations in numbered musical scores and staff scores using the key signature. For instance, Table 2.1 illustrates this conversion when the key signature is defined as 1 = C5, indicating that the numeral 1 in numbered musical scores corresponds to C5 in scientific pitch notation, which can be clearly represented in staff notation. Similarly, Table 2.2 demonstrates the conversion when the key signature is set as 1 = G5. Notably, in numbered musical scores, the pitch difference between 3 and 4, as well as between 7 and 1, is a semitone, while the difference between other adjacent notes is a whole tone (two semitones). In this way, it also conforms to the 12-tone equal temperament standard. Furthermore, the key signature in numbered musical scores is often represented in a manner such as “1=C" without specifying the octave. The interpretation of this notation as 1=C5, 1=C4, or other, depends on the specific performance context. Lastly, the rest note in the numbered musical score is represented as zero, denoted as 0.

Pitch in numbered musical notation	6	7	1	2	3	4	5	6	7	i	2
Pitch in scientific pitch notation	A5	B5	C5	D5	E5	F5	G5	A6	B6	C6	D6

Table 2.1. Pitch transformation between staff score and numbered musical score if 1 = C5

- **Duration of Notes:** Figure 2.6 depicts the representation of note durations in numbered musical scores, including both regular notes and rests.
- **Time Signature:** The time signature in numbered musical scores is similar to that in staff scores, consisting of a pair of numbers. The numerator denotes the number of beats per measure, while the denominator indicates the duration of each beat. For

Pitch in numbered musical notation	$\dot{6}$	$\dot{7}$	1	2	3	4	5	6	7	$\dot{1}$	$\dot{2}$
Pitch in scientific pitch notation	E5	F#5	G5	A5	B5	C6	D6	E6	F#6	G6	A6

Table 2.2. Pitch transformation between staff score and numbered musical score if 1 = G5

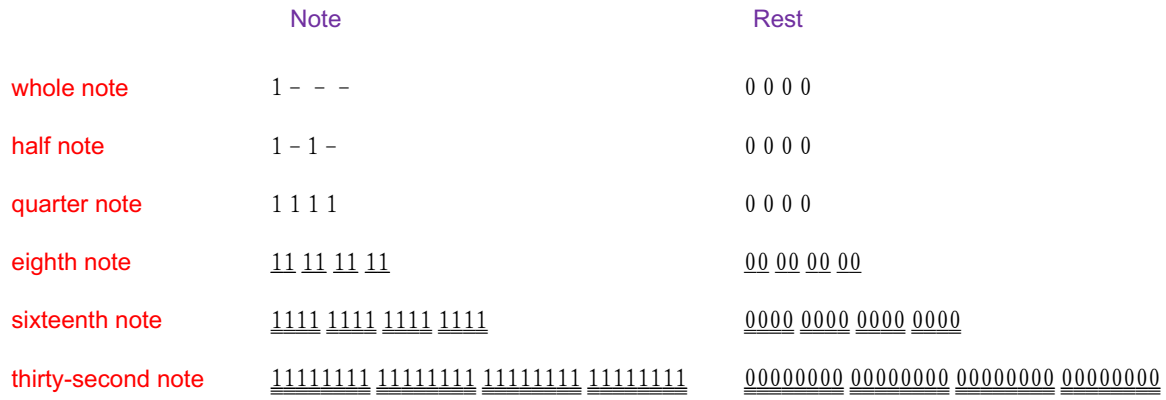


Fig. 2.6. Note durations in numbered musical scores

example, the time signature 2/4 in Figure 2.5 signifies that each beat is equivalent to the duration of a quarter note and each measure comprises 2 beats.

2.1.3. Gongchepu Score and Related Notations

A particular music notation, Gongchepu, was used in ancient China. As we will deal with ancient Chinese music represented in this notation, we describe it briefly. Figure 3.3 displays an example of a Gongchepu score. Figure 3.3(a) shows the original Gongchepu score example, and Figure 3.3(b) shows some corresponding explanations. In this score, it contains important musical information like the title, melody, lyrics, and punctuation. We can see from this example, that in Gongchepu, notes including pitch and duration are principally represented in traditional Chinese characters. Unlike staff scores and numbered musical scores that provide explicit specifications for both the pitch and duration of each note, Gongchepu leaves a degree of interpretational uncertainty. Different performers may adopt various interpretation methods. Our processing techniques for interpreting these scores, alongside a more detailed introduction including aspects of pitch, duration, time signature, and key signature, will be discussed in Chapter 3.

Despite its increased ambiguity compared to staff scores or numbered musical scores in certain respects, Gongchepu offers a flexible system, providing performers with ample creative space to improvise based on their personal understanding.

2.2. Background on Symbolic Music Generation

As discussed in Chapter 1, music generation can be broadly categorized into two types: symbolic music generation and acoustic music generation. Symbolic music generation concentrates on creating music in a symbolic form, such as musical score notation, contrasting with the generation of acoustic audio signals. Broadly, the tasks in symbolic music generation can be bifurcated into two categories: monophonic music generation and polyphonic music generation. The focus of this thesis lies within a subtask of symbolic monophonic music generation.

2.2.1. Monophonic Symbolic Music Generation

Monophonic music refers to the performance of a single melodic line at any given moment. An example of monophonic music in staff notation is illustrated in Figure 2.7. In some cases, the composition of monophonic music involves creating a melodious and pleasing musical line without any additional constraints. In other scenarios, the generation of a melody must adhere to specific constraints, such as those indicative of a particular style or emotion. This thesis focuses on the generation of melodies corresponding to lyrics, a topic that falls under the constrained monophonic melody generation.

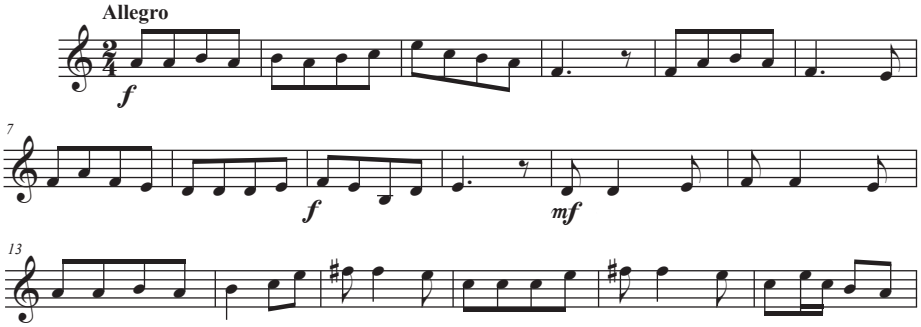


Fig. 2.7. A music excerpt illustrating a monophony melody. This excerpt comes from *Echigo-Jishi*, which is arranged by Y. Nagai and K. Kobatake.

2.2.2. Polyphonic Symbolic Music Generation

Contrary to monophonic music, polyphonic music implies the simultaneous play of multiple musical lines. In some cases, the multiple musical lines stem from the same timbre, such as depicted in Figure 2.8, where all musical lines originate from a single instrument, the piano. In other cases, the multiple musical lines arise from different timbres. An example of this is demonstrated in Figure 2.9, where the multiple musical lines are composed of the piano and the human voice.

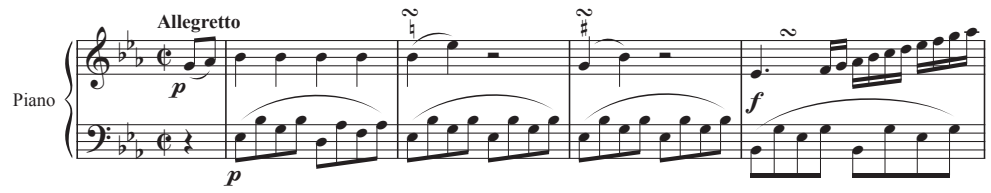


Fig. 2.8. An music excerpt illustrating polyphony melody from the same timbre. This excerpt comes from *An Chloe, K. 524*, composed by W. A. Mozart.

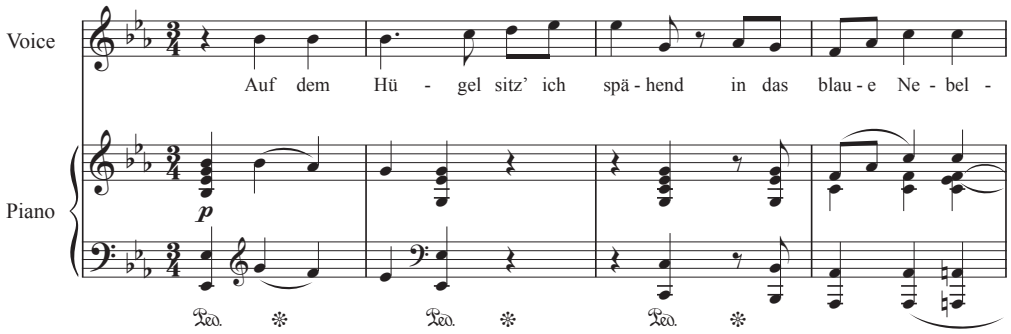


Fig. 2.9. A music excerpt illustrating polyphony melody from the same timbre. This excerpt comes from *An die ferne Geliebte, Op. 98*, composed by L. van Beethoven.

2.3. Related Work on Melody Generation from Lyrics

In this section, we introduce the related work concerning melody generation from lyrics. After the boom of deep learning, a majority of the research in this field leverage sequence deep models such as RNN [**45**] or the Transformer model. In the initial stage, research in this domain [**8, 39, 64**] usually employs RNN-based methods. Later studies [**53, 35, 66, 43, 24, 47**] commonly use Transformer architectures. As such, we will first revisit the RNN model and introduce RNN-based approaches utilized by previous research. We will then revisit the Transformer [**57**] model and introduce the Transformer-based approaches used in prior studies. Furthermore, since this thesis also handles non-popular music which has not been widely explored before, we will also introduce the music genres in previous research.

2.3.1. RNN Revisted and RNN-Based Approaches

RNN Revisted:

The Recurrent Neural Network (RNN) [**51**] constitutes a significant subclass of artificial neural networks. Because of its effectiveness in processing time series data, RNN is widely applied in different scenarios including sequence data, including but not limited to text generation, speech synthesis, music composition, speech recognition, and machine translation, etc.

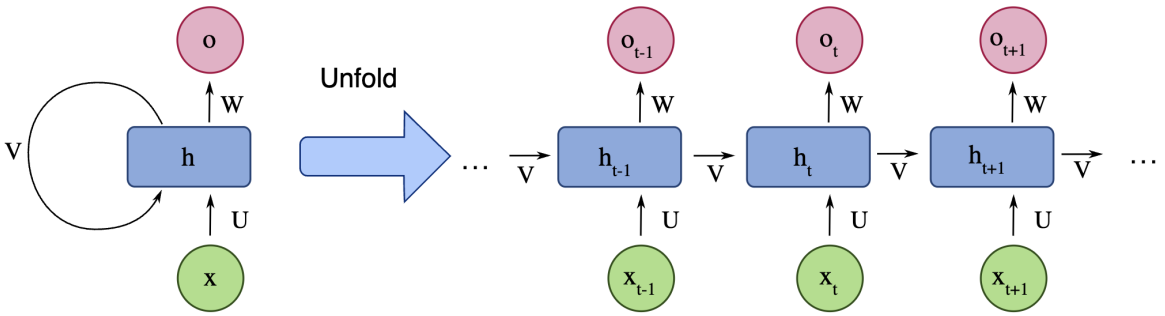


Fig. 2.10. The vanilla RNN architecture, referred from [**51**]

The simplest and the most original architecture of an RNN, often called a vanilla RNN, is shown in Figure 2.10. From this figure, it can be seen that at each time step t, the model accepts two inputs, x_t and h_{t-1}. x_t represents the current input, while h_{t-1} is a value accumulated from the previous $t-1$ time steps. On the output side, the model generates two results at each time step, o_t and h_t. o_t is the external output. h_t represents the accumulated output from time step 0 to time step t and is often referred to as the hidden state. The formulas between inputs and outputs at each time step can be derived as follows:

$$h_t = \sigma_h(Ux_t + Vh_{t-1} + b_h)$$
$$o_t = \sigma_o(Wh_t + b_o)$$

In the above formulas, U, V, and W are weight matrices, b_h and b_o are bias terms, and σ_h and σ_o are activation functions. As shown above, the RNN model contains internal hidden states that can store information from previous time steps, making it adept at handling sequential data. Furthermore, RNN has the advantage of parameter sharing. The weights U, V, W, b_h, and b_o are shared across time steps, which helps prevent an increase in model complexity and the number of parameters. With this setup, the size of the model does not increase even as the number of time steps does.

Although RNN has many advantages, it faces two significant challenges in the training process. The first challenge is the phenomena of vanishing and exploding gradients, which occur during the Back-Propagation Through Time (BPTT) [**59**] weight-updating process. These problems arise due to the repeated multiplication of gradients during backpropagation. If the gradients are smaller than 1, the resulting product can decrease exponentially, leading to vanishing gradients. Conversely, if the gradients are larger than 1, the product can increase exponentially, causing exploding gradients. Both situations render RNNs difficult to train effectively, particularly over long sequences. The second challenge, namely the issue of long-term dependencies [**12**], is closely tied to the problem of vanishing gradients. Due to vanishing gradients, weight updates become less effective as they propagate back in time,

making it difficult for RNNs to update parameters associated with early time steps in a sequence. As a result, RNNs struggle to retain and learn information from many time steps ago, making it hard for them to capture long-term dependencies in the data.

To address these issues, more advanced types of RNNs such as Long Short-Term Memory (LSTM) [**31**] and Gated Recurrent Unit (GRU) [**18**] have been proposed. These models introduce new mechanisms, called gates, which help manage the information flow in the network and allow for more efficient learning and remembering of information over longer sequences. In particular, these gates can selectively remember or forget information, which mitigates the problem of vanishing gradients or exploding gradients and allows the network to handle long-term dependencies more effectively.

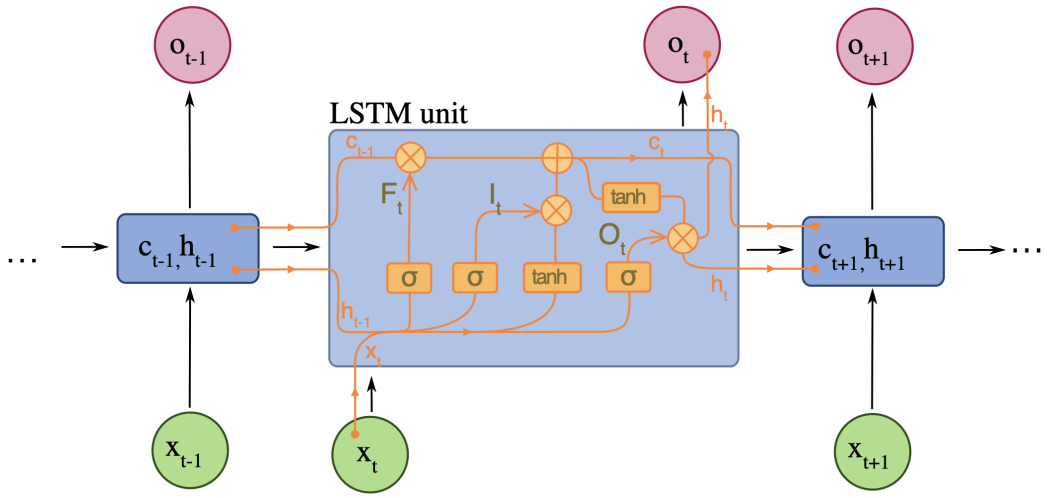


Fig. 2.11. The Long Short-Term Memory network architecture, referred from [**51**]

The illustration of Long Short-Term Memory (LSTM) is shown in Figure 2.11. Besides the hidden states h_t, LSTM also has the cell states c_t which do not exist in vanilla RNN. While h_t is used for short-term memory, c_t is used for long-term memory. Therefore, in LSTM, the input of the time step t includes x_t, c_{t-1} and h_{t-1}, the output includes o_t, c_t and h_t. The formulas between the inputs and outputs in LSTM are as follows:

$$
\begin{aligned}
i_t &= \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \\
f_t &= \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \\
o_t &= \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \\
g_t &= \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$

In the preceding formulas, $\odot$ signifies element-wise multiplication. W, and b are weight metrics and bias weight. The symbols i_t, f_t, and o_t correspond to the input gate, forget gate, and output gate, respectively, while c_t represents the current cell state, and g_t symbolizes the candidate cell state. The input gate i_t governs the amount of new information incorporated at the current step, while the forget gate f_t controls the quantity of information retained from the preceding steps, and the output gate o_t determines the degree of internal states that are outputted. Each of these gates yields values between 0 and 1 after the application of the sigmoid activation function. In essence, these three gates dictate whether the current input, the previous memory, and the present memory ought to be passed into the subsequent time step. Importantly, these gates alleviate the problems of vanishing and exploding gradients commonly associated with traditional RNNs when processing lengthy sequences, thereby enabling LSTM to tackle long-dependency issues.

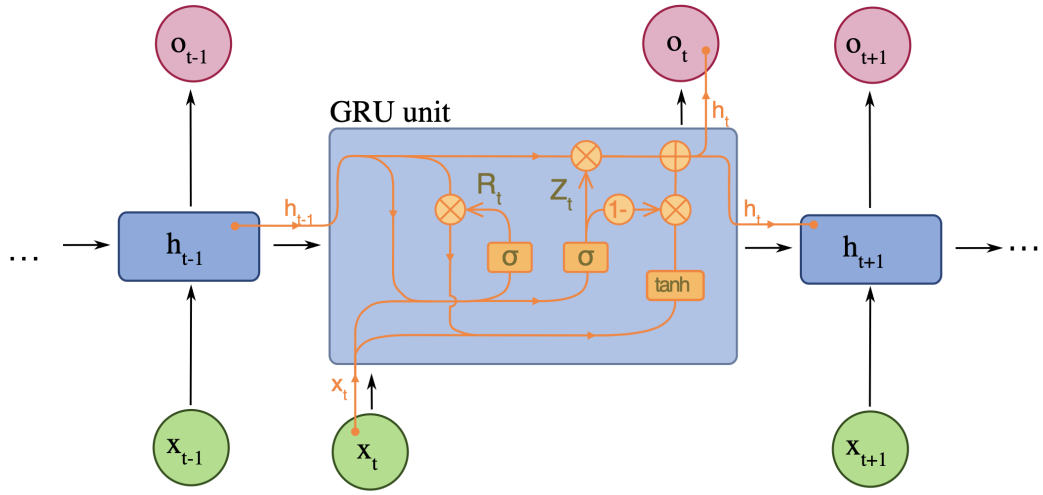


Fig. 2.12. The Gated Recurrent Unit architecture, referred from [**51**]

Another improved architecture of the original vanilla RNN is GRU. An illustration of GRU is shown in Figure 2.12. On the inputs and outputs side, GRU is similar to the vanilla

RNN, it does not contain the cell state c_t like LSTM. The inputs of GRU at time step t are x_t and h_{t-1}, the outputs are h_t and o_t ($o_t = h_t$ in GRU). The calculation in each step of GRU is shown below:

$$
\begin{aligned}
z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z) \\
r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r) \\
g_t &= \tanh(W_g x_t + U_g(r_t \odot h_{t-1}) + b) \\
h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot g_t
\end{aligned}
$$

In the aforementioned equations, W and b represent the weight matrices and bias vectors, respectively. The terms z_t and r_t correspond to the update and reset gates, respectively, while g_t and h_t represent the candidate hidden state and the current hidden state. The reset gate, r_t, determines the proportion of information from the preceding hidden states to be disregarded, or 'reset', in the computation of the candidate hidden state. Conversely, the update gate, z_t, dictates how much information from the previous hidden states should be retained in the current time step. Through the use of these gating mechanisms, the GRU model is capable of alleviating the vanishing gradient problem often associated with traditional RNNs.

RNN-Based Approaches

In earlier research on melody generation from lyrics using deep learning, RNN-based models were primarily employed [**8, 39, 64**]. For instance, the study in [**8**] utilizes both the RNN model and the GRU model. Specifically, they build the melody generation system based on an encoder-decoder framework, which is composed of two encoders, and a decoder. The encoders consist of GRUs and the decoder consists of RNNs. Besides, the works in [**39, 64**] employes LSTM models. Specifically, the study in [**39**] employs two separate LSTM encoder-decoder models; one is trained to predict the pitch sequence, while the other focuses on predicting the duration sequence. The work in [**64**] utilizes a conditional Long Short-Term Memory - Generative Adversarial Network (LSTM-GAN) for the melody generation from the lyrics task. This LSTM-GAN comprises an LSTM generator and an LSTM discriminator.

In general, compared to the Transformer model, RNN-based models tend to struggle with handling long-term dependencies. Music, being a type of sequential data, is more effectively represented by models that excel at managing these long-term dependencies. As such, current research in this field infrequently employs RNN as the foundational architecture, showing a preference for the Transformer architecture instead. In line with this trend, we also utilize the

Transformer model as our base architecture in this thesis. Subsequently, we will introduce the Transformer model and discuss the previously implemented Transformer-based approaches.

2.3.2. Transformer Revisted and Transformer-Based Approaches

Transformer Revisited

The Transformer model [**57**], is another important model for processing sequence data. Unlike the RNN model, which processes sequence data sequentially, the Transformer is capable of processing sequence data parallelly. This feature of parallel computation enables the Transformer to handle long-range dependencies more efficiently compared to RNNs.

The architecture of the Transformer is shown in Figure 2.13. The Transformer model utilizes an encoder-decoder architecture and integrates several key components that contribute to its performance. These key components include the self-attention mechanism, multi-head attention, a position-wise feed-forward network, and positional encoding. In the following sections, we will elaborate on these components one by one.

The first crucial component of the Transformer model is the self-attention mechanism. At an intuitive level, this mechanism allows each token in a sequence to determine the relevance or attention to other tokens in the sequence. Mathematically, the self-attention mechanism is defined as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$
$$Q = XW_Q$$
$$K = XW_K$$
$$V = XW_V$$

In the formulas above, Q, K, and V denote the query, key, and value vectors, respectively. W_Q, W_K, and W_V are the corresponding weight matrices. X denotes the input. d_k denotes the dimensionality of the key vectors. The query vector denotes the current token being considered, the keys represent all the tokens in this sequence, and the values are the actual values of each token. The final attention score output represents the degree of attention or relevance given by the query of the current token to each other tokens' key, determining how much the value each token contributes to the final output. Besides, the softmax function ensures the sum of the weights to 1 and are non-negative, and d_k is used as a scaling factor in the attention score calculation.

Relative to the self-attention mechanism, the Transformer model incorporates another technique known as the multi-head attention mechanism. Intuitively, this mechanism enables

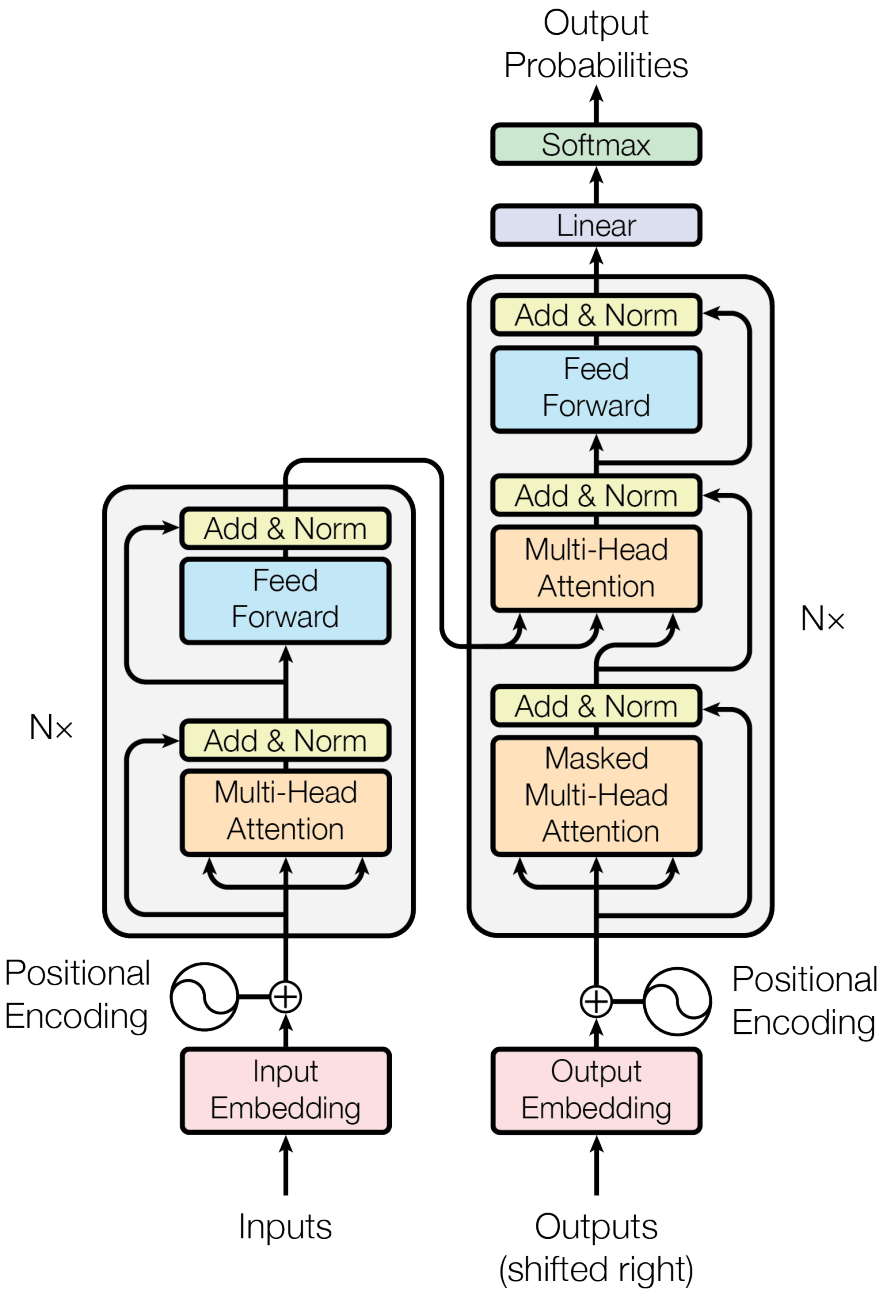


Fig. 2.13. The architecture of the Transformer model. Referred from [**57**].

the Transformer to capture information originating from different subspaces. Mathematically, the computations within the multi-head attention mechanism are as follows:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W_O$$
$$\text{where head}_i = \text{Attention}(QW_{Qi}, KW_{Ki}, VW_{Vi})$$

In the above formulas, W_O denotes the weight matrix, and h signifies the number of attention heads. After each head has generated its self-attention output, these outputs are concatenated. The resulting matrix is then further processed through a linear layer to yield the final output.

The next component is the position-wise feed-forward Network. In each layer of the encoder and decoder, there exists a position-wise feed-forward network (FFN). In each layer, the FFN is applied to each position with the same weights. But the weight differs in different layers. The mathematical formulas of the position-wise FFN are as follows:

$$FFN(x) = W_2 \cdot \mathrm{ReLU}(W_1 \cdot x + b_1) + b_2$$

In the above formula, x is the input, W_1 and W_2 denotes the weight matrices, and b_1 and b_2 are the bias. Overall, the FFN consisted of two linear transformations, and there is a ReLU activation function followed by the first linear transformation.

As we have previously mentioned, the Transformer model can perform parallel computations at each position. However, by doing so, it inherently lacks information about the sequence order. To solve this problem, the Transformer employs a positional encoding mechanism to encode the position information into the input. This is achieved by adding positional encodings to the input embeddings. The encoding values are defined by the following formulas:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$
$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

In the aforementioned formulas, pos signifies the positional information, while $2i$ and $2i+1$ denote the dimensions. The term d_{model} refers to the total dimension of the input embeddings, which is a predetermined hyperparameter. These positional encodings allow the model to discern the specific positions within the input sequence. Moreover, the choice of a sinusoidal function for the positional encoding is also meaningful. This sinusoidal function facilitates the model's ability to learn attention mechanisms based on relative positions. This is because for any fixed offset k, PE_{pos+k} can be represented as a linear function of PE_{pos}.

We have now completed the introduction to the Transformer model. The Transformer has demonstrated exceptional performance in a myriad of sequence tasks, such as machine translation, text generation, music generation, and so on.

Transformer-Based Approaches

After the first stage mainly using RNN-based models in the melody generation from lyrics task, people begin to focus on using Transformer-based models [**53, 35, 66, 24, 47**]. Specifically, In [**53**], the researchers design their SongMASS model by leveraging the foundation of the original Transformer architecture. The distinctive feature of their model compared to the original Transformer lies in their adoption of two separate encoders—one for encoding melody and the other for lyrics. Similarly, two distinct decoders are used to decode melody and lyrics individually. [**35**] introduces a melody generation system called TeleMelody based on a two-stage process. This process comprises a lyric-to-template module followed by a template-to-melody module. The term 'template' here refers to concatenated features such as tonality, chord progression, rhythm pattern, and cadence, which serve as intermediaries connecting lyrics to melodies. In each stage, a Transformer model is employed to facilitate the transformation. [**66**] introduces their models, termed ReLyMe, built upon the SongMASS and TeleMelody frameworks, thus also utilizing Transformer-based models. Building on the SongMASS and TeleMelody foundations, they integrate handcrafted rules about the relationships between lyrics and melodies to enhance melody generation performance. In [**24**], a model is presented that integrates the Transformer architecture with mutual information. The mutual information between lyrics and melody is utilized during training to ensure content consistency, while the Transformer model serves to extract semantic information from the lyrics. These adaptations contribute to the model's local interpretability. [**47**] generates melody based on the TeleMelody framework, so they also use the Transformer-based model. Similar to these above works, our thesis also uses the Transformer-based model.

2.3.3. Music Genres Explored in Existing Melody Generation from Lyrics Works

Since this thesis also handles non-popular music which has not been widely explored before, we also make a survey on the music genres in previous research, besides making a survey on the models used in the previous research. Works in [**64**], [**24**], and [**53**] primarily utilize popular music with English lyrics for their experimental datasets. Specifically, the dataset they use is publicly available online [**2**] and is created in [**64**], including 12,197 songs in total. Besides, [**8**] and [**39**] conduct their experiments using popular songs but with Chinese lyrics. Specifically, [**8**] crawls 18,451 Chinese pop songs from an online Karaoke app, and [**39**] collects 1,000 Chinese popular music pieces themselves. Additionally, [**35**] and [**66**] experiment on popular songs containing both English and Chinese lyrics. Specifically, in [**35**], in the template-to-melody stage, they use the dataset from [**4**], which is popular music. This stage determines the genre of the generated melody. [**66**] builds its models based on SongMASS and TeleMelody and follows the data collection pipeline of SongMASS and TeleMelody. Therefore, this research also aims at generating popular music.

However, as mentioned above, most of the current research related to melody generation from lyrics is about generating popular music. Studies on traditional Chinese music are scarce, with only two notable contributions from [**40**] and [**47**]. The lyrics in these traditional pieces largely come from classical Chinese poetry. In their research, [**40**] uses the Conditional Random Field method [**38**] to compose melodies from classical Chinese poetry on a small dataset. They rely solely on traditional machine learning models and train on a small paired dataset (about 100 songs). [**47**] applies a deep learning model, TeleMelody [**35**], on an unpaired dataset through a two-stage process: lyric-template and template-melody. However, they don't establish a direct link between lyrics and melody since they don't use an end-to-end modeling method and only utilize unpaired data. In summary, in the realm of traditional Chinese music research, neither study leverages deep learning to directly connect lyrics and melody. In this thesis, we also partly focus on traditional Chinese music. However, unlike the two aforementioned studies, we use deep learning to directly model the lyrics-melody relationship with a much larger paired dataset (about 6500 songs). In addition, we also investigate the use of additional linguistic and stylistic features in music generation.

Chapter 3

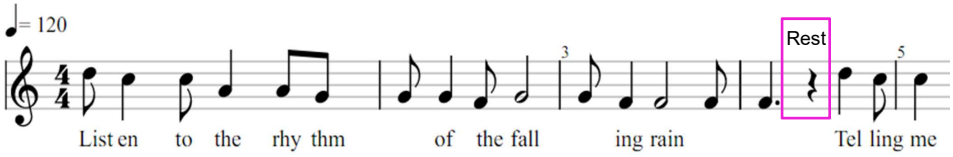
Dataset, Data Preprocessing, and Data Pepresentation

This chapter describes in detail the necessary data processing before it can be used to train a model. Firstly, we introduce the datasets used in this thesis, which include two kinds of song datasets. Then, we introduce how we preprocess the two kinds of data, respectively. Finally, we introduce how to represent the preprocessed data.

3.1. Dataset

We use two different datasets. One dataset is the melody with English lyrics, The other is the melody with Chinese lyrics.

3.1.1. Dataset with English Lyrics



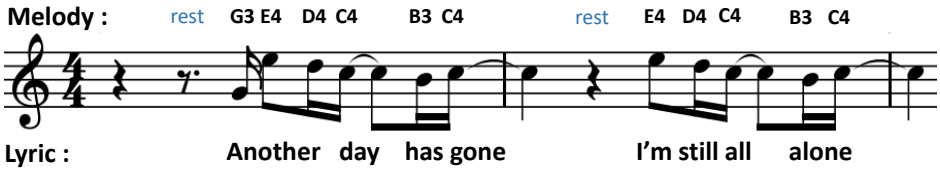
Lyrics	List	en	to	the	rhy	thm	of	the	fall	ing	rain	Tel	ing	me
Note	D5	C5	C5	A4	A4	G4	G4	F4	G4	F4	F4	D5	C5	C5
Duration	0.5	1	0.5	1	0.5	1	1	0.5	2.5	1	4	1	0.5	1
Rest	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fig. 3.1. A syllable-note alignment example of an English song excerpt from [**2**]. This figure is taken from [**64**].

The first song dataset used in this thesis consists of melodies and English lyrics. We obtained the song dataset with English lyrics from [**64**], which is publicly available online [**2**]. This dataset comprises a total of 12,197 MIDI files, with 7,998 files obtained from the

LMD-full MIDI Dataset [**4**], and the remaining 4,199 files sourced from the Reddit MIDI dataset [**5**]. The style of this dataset belongs to popular music.

An example excerpt of a song from this dataset is presented in Figure 3.1. In the English language, a word is composed of several syllables, and in this dataset, each syllable corresponds to a note. Each note includes two fundamental attributes: pitch and duration. Additionally, each syllable is associated with a Rest attribute, indicating the presence or absence of a Rest note. In summary, each syllable in the dataset is associated with three musical attributes: note (pitch), duration, rest. Furthermore, besides the alignment of syllables with notes, this dataset also provides alignment between words and notes. An illustration of the alignment between words and notes can be observed in Figure 3.2. In this thesis, we adopt the word-to-note alignment data following the approach outlined in [**53**]. In this example, each word is aligned with a subsequence of notes.



Paired Aligned Data :

Lyric	**Another**				**day**	**has**	**gone**	**I'm**		**still**	**alone**	
Pitch	R	G3	E4	D4	C4	B3	C4	R	E4	C4	B3	C4
Duration	$\frac{7}{16}$	$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{3}{16}$	$\frac{1}{16}$	$\frac{5}{16}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{3}{16}$	$\frac{1}{16}$	$\frac{5}{16}$

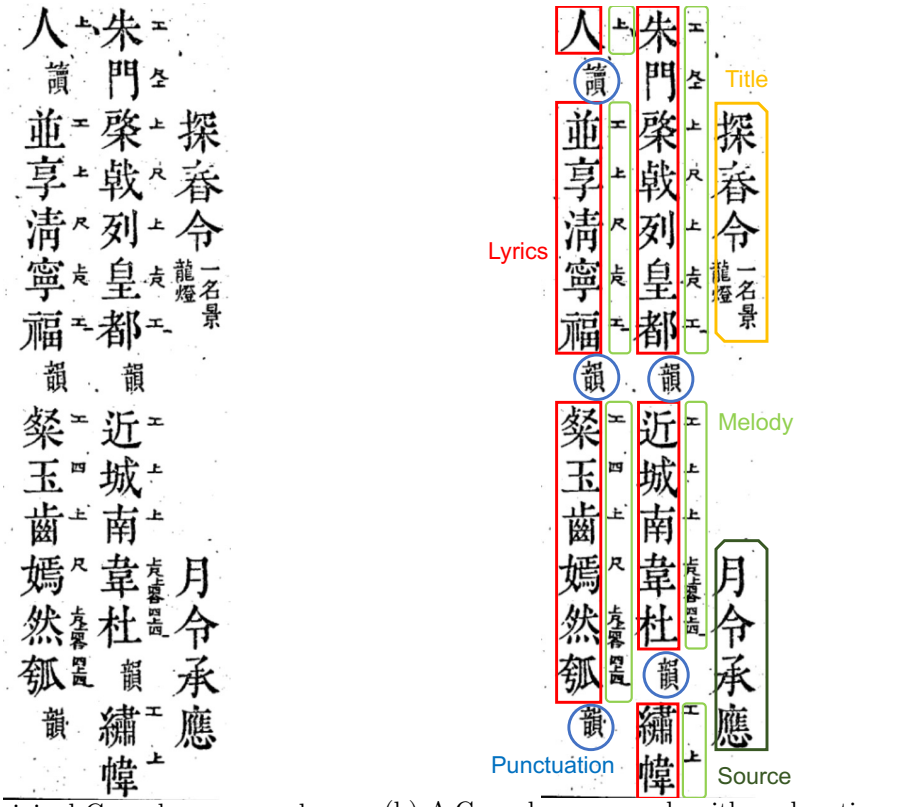
Fig. 3.2. A word-note alignment example of an English song excerpt from [**2**]. This figure is pasted from [**53**].

3.1.2. Dataset with Chinese Lyrics

The second song dataset used in this thesis consists of melodies and Chinese lyrics. Instead of using the Chinese lyrics dataset from modern popular music, we use the song dataset from ancient times. Specifically, the dataset is from `Nine Modes Manual Online` [**52**]. This dataset is created by Casey Schoenberger and stored in the Digital Collections Portal at Pao Yue-kong Library. This online dataset is digitized from "one of the oldest and most comprehensive collections of Chinese vocal melodies in existence, `Jiugong Dacheng Nanbei Ci Gongpu` [**65**]." (According to [**52**]). The lyrics in this dataset include classical Chinese poetry and some classical Chinese operas (which are seen as variants or extensions of classical Chinese poetry). The dataset contains approximately 6500 songs. This dataset is used for the first time in machine learning tasks in this thesis. Compared to previous research, [**40**] utilized around 100 paired songs between classical Chinese poetry and melody. Meanwhile, [**47**] trained a model on over 6000 songs, but these songs only contained melodies

without lyrics. The dataset employed in this thesis represents a significantly larger paired dataset.

One reason for using this dataset is that non-popular music is rarely considered in previous research. Another motivation stems from historical context: many classical Chinese poems could be sung in ancient times, but many associated melodies have been lost, leaving only the poetry itself. Given the assumption that the lost melodies share similar elements, such as styles and genres, with the preserved melodies. If we can use deep neural networks to model the remaining melodies and their corresponding poems, we can potentially contribute to the restoration of these lost melodies.



(a) A original Gongchepu example (b) A Gongchepu example with explanations

Fig. 3.3. A Gongchepu example from `Jiugong Dacheng Nanbei Ci Gongpu`

An example of a song from the original book, `Jiugong Dacheng Nanbei Ci Gongpu`, is displayed in Figure 3.3. This song is documented in a traditional Chinese score called Gongchepu. Within Gongchepu, both lyrics and melody are primarily represented in traditional Chinese characters, with each lyric corresponding to one or more notes. After digitalization, each Gongchepu score is summarized in a table. For example, part of the Gongchepu shown in Figure 3.3 is digitized in a table like Table 3.1. This example is actually a song type in this dataset, which is called Sanban. Sanban scores do not include rhythm symbols (_ is just a symbol that represents the end of a sentence, and it does not appear in every

song). We also present another example which is from non-Sanban scores in Figure 3.5(a). Parts of this non-Sanban score example can be digitized in Table 3.2.

In addition to the information on lyrics, melody (notes), and sentence number (the corresponding sentence in the song) in these tables, we also obtain the meta-information about the title, source, and Gongdiao (to be introduced in more detail later) of each song from the online dataset.

Lyric	Note	Sentence No.
朱 (Zhu)	工	1
門 (Men)	尺上	1
棨 (Qi)	上	2
戟 (Ji)	尺	1
列 (Lie)	上	1
皇 (Huang)	上尺	1
都 (Du)	工_	1
近 (Jin)	工	2
城 (Cheng)	上	2

Table 3.1. Example of Figure 3.3's song after digitization

Lyric	Note	Sentence No.
咱 (Zan)	、六	1
疑 (Yi)	◦五仩五	1
惑 (Huo)	□六	1
忽 (Hu)	、工	1
然 (Ran)	□尺工	2

Table 3.2. Example of Figure 3.5's song after digitization

3.2. Chinese lyrics Data Preprocessing

For the Chinese lyrics dataset we used, the symbols in traditional Gongchpu have a big difference from the modern Western staff scores, so we need to do some data preprocessing operations to interpret the traditional Gongchepu score into staff scores. In this section, we first introduce pitch preprocessing and duration preprocessing, respectively. Then, we can transform the traditional Gongchepu into staff. For the English lyrics dataset whose scores are already in staff, it does not need such a transformation step.

3.2.1. Pitch Preprocessing

The transformation rules between pitch in Gongchepu and pitch in numbered musical notation are proposed in [**63**]. We illustrate the process in the first two rows in Table 3.3

and Table 3.4. These two notations can only reflect relative pitch instead of absolute pitch. Absolute pitch for each pitch from Gongchepu can only be obtained when, for example, one pitch from Gongchepu is specified to correspond to a pitch in scientific pitch notation (e.g., 上 = C5 or 1 = C5). However, such key signature information (1 = C, meaning C major) is unclear in this dataset. Gongdiao in the meta-information specifies key signature. However, it is unclear how to align each Gongdiao category to a specific key in Western music. Therefore, assuming the standard correspondence 上 = C5 (i.e., C major), as shown in the last row of Table 3.3 and Table 3.4, we then transform pitches in Gongchepu into pitches in scientific pitch notation.

Pitch in Gongchepu score	合	四	一	**上 (std)**	尺	工
Pitch in numbered musical notation	$\underset{\cdot}{5}$	$\underset{\cdot}{6}$	$\underset{\cdot}{7}$	1	2	3
Pitch in scientific pitch notation	G4	A5	B5	C5	D5	E5

Table 3.3. Pitch transformation when seeing 上 as the standard (Part 1). This table shows part of the pitches used in the dataset.

Pitch in Gongchepu score	凡	六	五	乙	仩	伬	仜
Pitch in numbered musical notation	4	5	6	7	$\dot{1}$	$\dot{2}$	$\dot{3}$
Pitch in scientific pitch notation	F5	G5	A6	B6	C6	D6	E6

Table 3.4. Pitch transformation when seeing 上 as the standard (Part 2). This table shows the remaining pitches used in the dataset.

3.2.2. Duration Preprocessing

Gongchepu scores can be divided into two categories based on the presence of rhythm symbols. Scores without any rhythm symbols are called Sanban, which means the scores have free rhythm, similar to Senza Misura in Western music. The performance of Sanban scores heavily relies on the improvisation of the performer. For notes in Sanban scores, since there are no specific rhythm symbols to determine the duration, we simply set the duration of each note as $\frac{1}{2}$ quarter lengths.

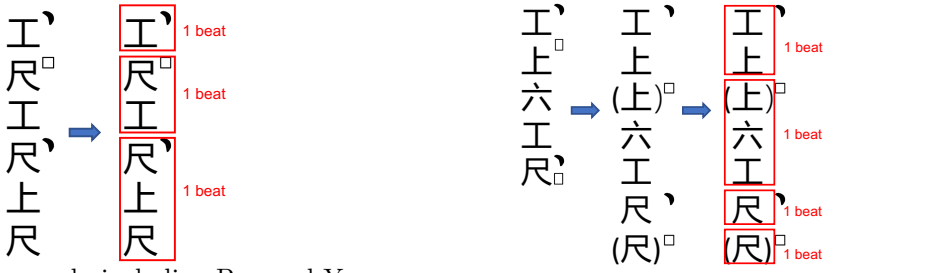
The other category is the non-Sanban score, which contains some rhythm symbols. Table 3.5 displays the rhythm symbols [**52**] used in the non-Sanban scores of `Jiugong Dacheng Nanbei Ci Gongpu`. Rhythm symbols in Gongchepu are all attached to the right of the pitch. There are six different types of rhythm symbols in total. Ban, Yan, and Zengban mean that the attached notes are at the beginning of a new beat. Yaoyan, Yaoban, and ZengYaoban indicate that the duration of the attached note should extend to the next beat.

Aside from these distinctions on the beat, they differ only in dynamics in performance. Since our research focuses solely on the score level, their differences in dynamics are ignored.

Symbol	CH	CH-Pinyin	Description
◝	板	Ban	Begin of a beat
□	眼	Yan	
◟	贈板	Zengban	
▯	腰眼	Yaoyan	Beyond a beat
∟	腰板	Yaoban	
⌞	贈腰板	ZengYaoban	

Table 3.5. Rhythm symbols in the used dataset. Notice the difference between □ and ▯

We now explain how to divide beats based on these rhythm symbols in detail. An example of dividing beats in a Gongchepu score containing only Ban, Yan, or Zengban is illustrated in Figure 3.4(a). If the score includes Yaoyan, Yaoban, or ZengYaoban, as depicted in Figure 3.4(b), we first transform it into a score containing only Ban and Yan, according to [**62**]. Then, we divide the beats using the same way as Figure 3.4(a).



(a) Gongchepu only including Ban and Yan

(b) Gongchepu including Yaoyan or Yaoban

Fig. 3.4. The explanation of dividing beats

However, the time signature and the assignment of durations to each note within a beat are not explicitly specified in each Gongchepu score, which are mandatory elements in the staff. As a result, we establish rules to define these aspects. First, for the time signature, we consider it as 4/4 (In Gongchepu, one Ban followed by three Yans usually signifies 4/4, while one Ban followed by one Yan indicates 2/4. However, this information is unclear in this dataset, so we simply consider all as 4/4). For situations involving multiple notes in a beat, we follow the methods described in [**62**]:

(1) If one beat includes two notes, we assign the duration of each note as $\frac{1}{2}$ beat.

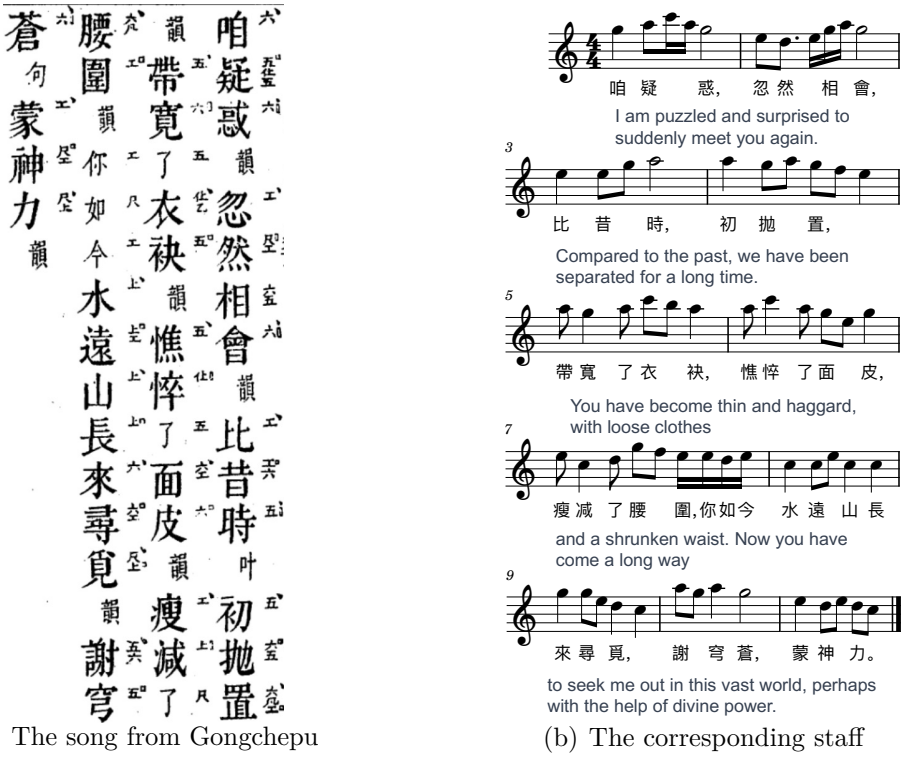
(2) If one beat includes three notes, we assign the duration of the first note as $\frac{1}{2}$ beat and the last two notes both as $\frac{1}{4}$ beat.

(3) If a beat consists of four notes, then each note occupies $\frac{1}{4}$ of the beat.

(4) If a beat consists of more than four notes, we assign the duration of each note to $\frac{1}{2}$ quarter length.

Now, we finish the duration transformation process.

3.2.3. A Entire Transform Example from Gongchepu to Staff Score



(a) The song from Gongchepu (b) The corresponding staff

Fig. 3.5. An example from Gongchepu to staff

After pitch transformation and duration transformation, we now can transform the Gongchepu into the staff score. Figure 3.5 shows an example from the original Gongchepu to staff notation after the pitch and duration transformation introduced before.

3.3. Data Representation

After introducing data preprocessing methods, we now introduce how we represent the data. In this thesis, for both the English lyrics dataset and Chinese lyrics dataset, following approaches similar to those in [**35, 53, 66**], we represent both lyrics and melody in symbolic sequences. For the Chinese lyrics dataset, for instance, we represent the lyrics and melody of the first two sentences in Figure 3.5 as shown in Figure 3.6. For the English lyrics, we also represent them in similar ways.

For the lyrics, [*sep*] is used to separate two sentences, and [*align*] is used to separate two Chinese characters or two English words. For the melody, the pitch is directly transformed into MIDI numbers ranging from 0 to 127, and the number of the rest note is set to 128. For the duration, in the Chinese lyrics dataset, the following formula is employed to convert:

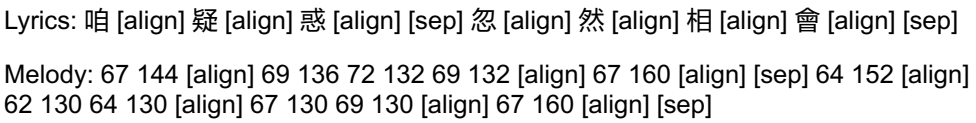


Fig. 3.6. Data Representation Example of Chinese lyrics song

$$d_{model} = 16 \times d_{qr} + 128$$

.

In this formula, d_{qr} represents the quarter length of a note, and d_{model} refers to how it is represented as the input of the model. The reason for adding 128 is to avoid overlapping between the number range of duration and the pitch range since the pitch range is from 0 to 128. And the reason for multiplying 16 is to obtain an integer. For example, if the duration of a note is 1 quarter length, it is represented as $d_{model} = 16 \times 1 + 128 = 144$. If a note is a quaver, it is represented as $d_{model} = 16 \times \frac{1}{2} + 128 = 136$. Additionally, $[sep]$ and $[align]$ are also inserted into the melody. For the English lyrics dataset, we directly use the methods in [**53**], which also map the duration into an integer greater than 128. However, it has a slight difference from the above covert method: [**53**] maps all durations into continuous integers from 129 to 144 by directly creating a vocabulary directory.

Chapter 4

Melody Generation System and Training Method

In this chapter, we introduce the melody generation system and the training method. We first introduce the basic framework and training method, then we introduce the adaptations we make based on the basic framework, including incorporating tones in Chinese, incorporating part-of-speech embeddings, and re-ranking generation candidates based on Gongdiao-style.

4.1. Basic Framework and Traning Method

In this section, we first introduce the base model we adopt, which is seen as a baseline and the basic model. Then, we introduce the training method.

4.1.1. Basic Model Framework

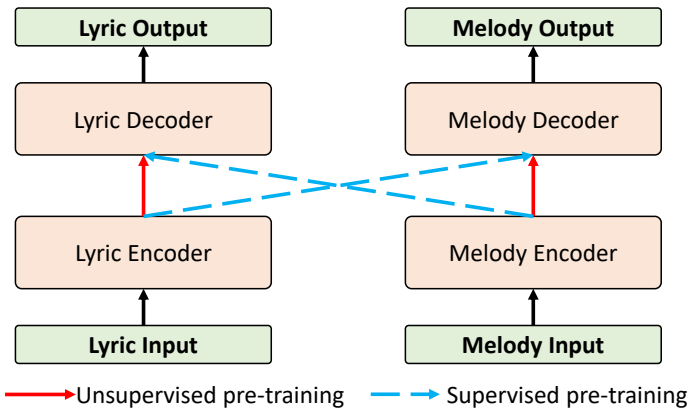


Fig. 4.1. The basic model framework. This figure is pasted from [**53**]

In this paper, we adopt the basic architecture from SongMASS [**53**], which is shown in Figure 4.1. SongMASS is a Transformer-based framework. This model comprises four

components: a lyrics encoder, a lyrics decoder, a melody encoder, and a melody decoder. We chose this model as our base model because it has been shown to perform well in the one-to-many (one lyric is corresponding to one or multiple notes) melody generation from the lyrics task.

From Figure 4.1, we can also see the general pretraining method which includes both unsupervised and supervised pretraining. The unsupervised pretraining takes place within lyric-to-lyric and melody-to-melody, helping the model to grasp lyrics and melody respectively. The supervised pretraining is applied in both lyric-to-melody and melody-to-lyric directions, which is to learn a shared space between lyrics and melody.

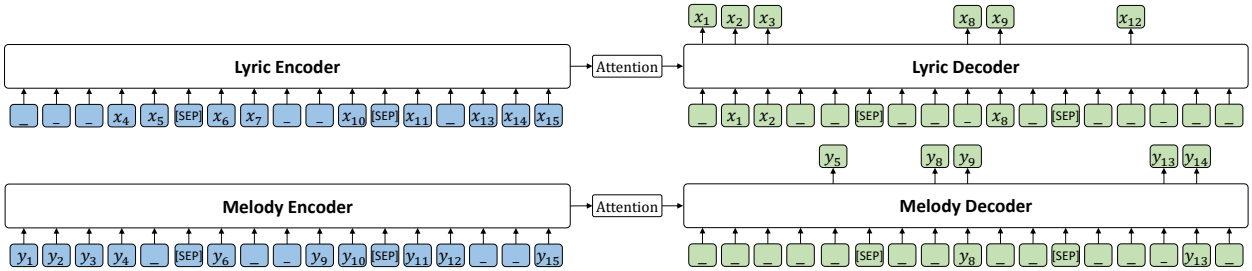


Fig. 4.2. The unsupervised pretraining in the SongMASS model. This figure is pasted from [**53**]

Then, we present detailed formulas for both unsupervised pretraining and supervised pretraining. For unsupervised pretraining, SongMASS adopts a masking strategy similar to the one used in the original MASS model [**55**]. Specifically, the original MASS model randomly selects a segment of tokens and replaces them with mask tokens. This masked sequence then serves as the input to the encoder, while the decoder predicts the masked tokens. SongMASS applies this masking strategy to each sentence in the song, as illustrated in Figure 4.2. To clarify with formulas: Let $\mathcal{X}$ represent the corpus of the lyrics and $\mathcal{Y}$ represent the corpus of the melodies. Given that, x is an element of $\mathcal{X}$ and y is an element of $\mathcal{Y}$. Let θ^{enc} and θ^{dec} denote the parameters of the encoder and the decoder, respectively. If S represents the number of sentences in a sequence x and $u_i : v_i$ indicates the masked tokens in the i-th sentence where u_i represents the first token of this masked segment and v_i represents the last token of this segment, then $x^{\setminus\{u_i:v_i\}_{i=1}^{S}}$ represents the masked song-level sequence. The loss for the lyric-to-lyric pretraining can be expressed as follows:

$$L_x = L\left(\mathcal{X}; \theta_x^{enc}, \theta_x^{dec}\right) = \sum_{x \in \mathcal{X}} \sum_{i=1}^{S} \log P\left(x^{u_i:v_i} \mid x^{\setminus\{u_i:v_i\}_{i=1}^{S}}; \theta_x^{enc}, \theta_x^{dec}\right)$$

Similarly, the loss formula for melody-to-melody pretraining can be described as follows:

$$L_y = L\left(\mathcal{Y}; \theta_y^{enc}, \theta_y^{dec}\right) = \sum_{y \in \mathcal{Y}} \sum_{i=1}^{S} \log P\left(y^{u_i:v_i} \mid y^{\setminus\{u_i:v_i\}_{i=1}^{S}}; \theta_y^{enc}, \theta_y^{dec}\right)$$

Next, for the lyric-to-melody supervised pretraining, the loss formula can be expressed as follows:

$$L_{xy} = L\left(\mathcal{X}, \mathcal{Y}; \theta_x^{enc}, \theta_y^{dec}\right) = \sum_{(x,y)\in(\mathcal{X},\mathcal{Y})} \log P\left(y \mid x; \theta_x^{enc}, \theta_y^{dec}\right)$$

For the melody-to-lyric supervised pretraining, the loss formula can be expressed as follows:

$$L_{yx} = L\left(\mathcal{Y}, \mathcal{X}; \theta_y^{enc}, \theta_x^{dec}\right) = \sum_{(y,x)\in(\mathcal{Y},\mathcal{X})} \log P\left(x \mid y; \theta_y^{enc}, \theta_x^{dec}\right)$$

Finally, the total loss for the pretraining stage is:

$$L_{pre} = L_x + L_y + L_{xy} + L_{yx}$$

Now we finish the introduction about the basic framework and the general pretraining method. Next, we will introduce the specific training methods for both the Chinese lyrics dataset and the English dataset.

4.1.2. Training Methods for the Chinese lyrics dataset

For the Chinese lyrics dataset, the whole training process is divided into two stages: the pretraining stage and the fine-tuning stage. During the pretraining stage, we use all the data for training. This stage consists of four parts: lyrics-lyrics pretraining, melody-melody pretraining, lyrics-melody pretraining, and melody-lyrics pretraining. In the fine-tuning stage, we only use Non-Sanban scores for training. Although Sanban scores leave room for improvisation to performers, it is not useful for machine learning models to learn rhythm information solely from Sanban scores since the durations of almost all notes in Sanban scores are the same at the score level. As a result, we only use Non-Sanban scores to fine-tune the model, thereby increasing the probability of generating melodies with more diverse rhythms. The fine-tuning stage involves only lyrics-melody training.

4.1.3. Training Methods for the English lyrics dataset

For the English lyrics dataset, since it does not contain Sanban data like the Chinese lyrics dataset, we directly use one-stage training, which is composed of lyrics-lyrics training, melody-melody training, lyrics-melody training, and melody-lyrics training.

4.2. Incorporating POS and Tone Embeddings

In this section, we introduce the first adaptations we adopt based on the basic architecture, incorporating Part-of-Speech embeddings and tone embeddings. The POS embedding adaptations are implemented in both the Chinese lyrics dataset and the English lyrics dataset; but the tone embedding adaptations are only applied to the Chinese lyrics dataset

since English is a non-tonal language (Tonal language means for each pronunciation, different tones can represent different words). We first introduce the motivations for using these two kinds of information, then we introduce our incorporating methods.

4.2.1. Motivations for Incorporating POS Embeddings

We first introduce the motivation for incorporating POS Embeddings. In natural language, when we speak a sentence, there are potential prosodic boundaries between words, which help to highlight language structures and make the language easier to understand. For example, consider the following sentence:

"I enjoy reading books and playing games."

When we speak the above sentence which does not contain punctuation marks inside the sentence (excluding the period in the end), we actually usually say it as the following:

"I enjoy reading books (short pause) and (short pause) playing games."

That is, there might be potential subtle pauses or prosodic boundaries between some words, which help to emphasize the structure of the sentence. Such potential prosodic boundaries exist in any language. Also, this situation is similar when singing lyrics. The rhythm in the melody also tends to naturally align with prosodic units, making the lyrics easier to comprehend. Therefore, to make the model learn the prosodic boundaries information, we incorporate POS into the model since POS is highly related to prosodic boundaries. This adaptation is applied both to the English lyrics dataset and to the Chinese lyrics dataset.

4.2.2. Motivations for Incorporating Tone Embeddings

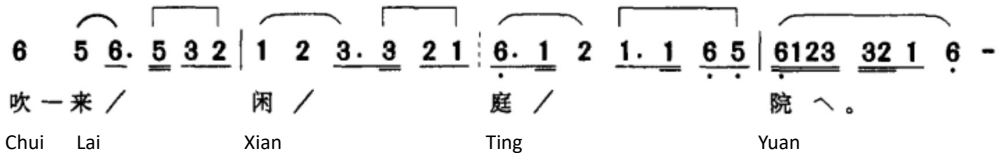


Fig. 4.3. An example showing how pitches in melody related to tones in lyrics from [**25**]

Besides, we also incorporate tone embeddings. English is a non-tonal language, meaning that the basic meanings of words do not rely on fixed pitch patterns. Although English indeed exists tonal variation, which can convey emotion, pose questions, or add emphasis, they do not alter the fundamental meanings of the words. In contrast, Chinese is a tonal language featuring four tones in its characters. As indicated in [**25**], the pitch flow of a melody in traditional Chinese opera music is usually closely related to the tones of the corresponding lyrics. Figure 4.3 provides an example illustrating this kind of relationship. In this figure,

the melody is represented in numbered musical notation, with pitch explanations detailed in Table 3.3 and Table 3.4. Each Chinese character in the figure is accompanied by a trend line that depicts the pitch change of the character's tone, which closely resembles the pitch change of the corresponding melody. This phenomenon is common in traditional Chinese opera, so it may be useful to incorporate this characteristic into the model for the Chinese lyrics dataset.

4.2.3. Methods to incorporate Tone and POS Embeddings

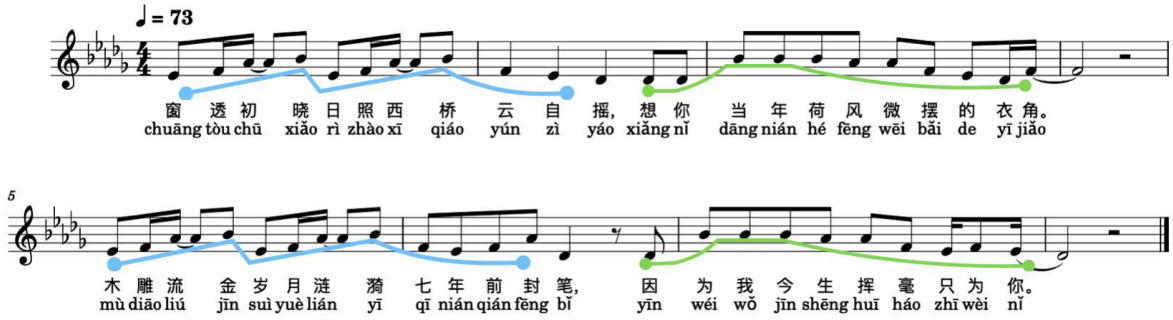


Fig. 4.4. An example showing the lyrics with the same structures tend to have similar pitch flows

In previous research on generating melody from lyrics, there have been attempts to incorporate such features. For instance, when generating the melody's pitch, [**40**] uses tone as the input feature, but it ignores the original word and solely considers tones. One reason is they use the Conditional Random Field as the model to train on a small dataset of about 100 songs. Using the original word as a feature can make the model difficult to fit. [**66**] employs the original word information, tone information, and POS information, and designs manual rules to exploit them. For instance, in terms of tone, when a lyric corresponds to several notes, if the tone change of this lyric aligns with the pitch changes of the corresponding notes, then such notes are more likely to be generated. As another example relating to POS information, as shown in Figure 4.4, if two lyric segments have the same structure (can be judged by extracting the POS information from the lyrics), the melodies with similar pitch flows are more probable to be generated. However, there are several disadvantages to using manually specified rules. First, these rules are difficult to generalize. When dealing with a different type of music, it is necessary to create new, personalized rules, which require expertise and can be time-consuming. Second, even within the same type of music, manually specified rules may not accommodate all situations, as they are based on experiential rules, and numerous situations could violate these rules. Therefore, a better way to exploit the information is to let the model learn to encode the information automatically for the purpose of melody generation.

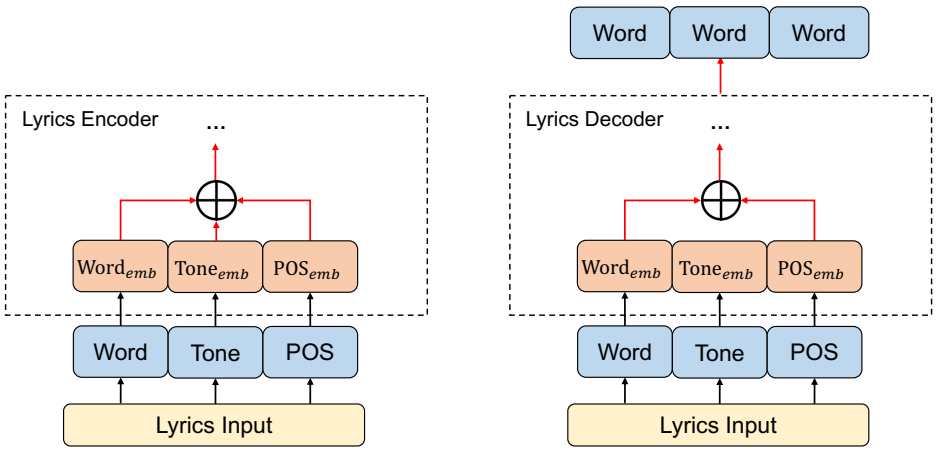


Fig. 4.5. Incoporating Tone Embedding and Part-of-Speech Embedding into lyrics encoder and lyrics decoder

Figure 4.5 provides a general illustration of our approach. Broadly speaking, we retain the original structure of the melody encoder and the melody decoder, focusing our modifications on the lyrics encoder and decoder components. We incorporate both tone embedding and POS (Part-of-Speech) embedding into these parts. Specifically, the input to both the encoder and the decoder is a concatenation of the word embedding sequence, tone embedding sequence, and POS embedding sequence. To match this, we triple the word sequence at the output of the decoder. During the lyrics-melody and melody-lyrics pretraining processes, the tone and POS of the words are directly extracted. To describe the inputs and outputs of these processes in mathematical terms, for the lyrics-melody process, assume there's a sequence of lyrics with a length of 5, excluding the [SEP] symbols. Let c be an intermediate variable representing the encoded context derived from the input data. This process can be represented as follows:

$$c = \text{Lyrics\_Encoder}(x_1, x_2, x_3, x_4, x_5; \text{tone}(x_1), \text{tone}(x_2), \text{tone}(x_3), \text{tone}(x_4), \text{tone}(x_5); \\ \text{pos}(x_1), \text{pos}(x_2), \text{pos}(x_3), \text{pos}(x_4), \text{pos}(x_5))$$

$$\text{Melody\_Decoder}(c; \_, y_1, y_2, y_3, y_4) = (y_1, y_2, y_3, y_4, y_5)$$

Similarly, the melody-lyrics process can be represented using the formulas below:

$$c = \text{Melody\_Encoder}(y_1, y_2, y_3, y_4, y_5)$$

$$\text{Lyrics\_Decoder}(c; \_, x_1, x_2, x_3, x_4; \_, \text{tone}(x_1), \text{tone}(x_2), \text{tone}(x_3), \text{tone}(x_4); \_, \text{pos}(x_1), \text{pos}(x_2), \\ \text{pos}(x_3), \text{pos}(x_4)) = (x_1, x_2, x_3, x_4, x_5; x_1, x_2, x_3, x_4, x_5; x_1, x_2, x_3, x_4, x_5)$$

In the lyrics-lyrics pretraining process, the words are partially masked. We utilize the tone and POS of the original non-masked words. Representing this process with formulas, and

assuming x_2 and x_3 are the masked tokens, the representation is as follows:

$$c = \text{Lyrics\_Encoder}(x_1, \_, \_, x_4, x_5; \text{tone}(x_1), \text{tone}(x_2), \text{tone}(x_3), \text{tone}(x_4), \text{tone}(x_5); \\ \text{pos}(x_1), \text{pos}(x_2), \text{pos}(x_3), \text{pos}(x_4), \text{pos}(x_5))$$

$$\text{Lyrics\_Decoder}(c; \_, \_, x_2, \_, \_; \_, \text{tone}(x_1), \text{tone}(x_2), \text{tone}(x_3), \text{tone}(x_4); \\ \_, \text{pos}(x_1), \text{pos}(x_2), \text{pos}(x_3), \text{pos}(x_4)) = (x_2, x_3; x_2, x_3; x_2, x_3)$$

By incorporating these embeddings, the model can learn to capture tone information, POS information, and the original word information automatically without the need for manual rules. Note that we only incorporate the POS embeddings for the English lyrics dataset.

4.3. Re-ranking Generation Candidates Based on Style during the Inference Stage

In this section, we introduce another adaptation - adding style constraints. Since the style labels only exist in the Chinese lyrics dataset, we only apply them in the Chinese lyrics dataset. The purpose is to generate melodies corresponding to the specific styles of Chinese lyrics. We first introduce our motivations, then we introduce our specific methods.

4.3.1. Motivations to Utilize Gongdiao-Style

Besides the tone and POS features, we aim to integrate some global features into the model to constrain the generation process. One feature we consider is style. In the meta-information of the Chinese lyrics dataset, there exist Gongdiao labels, which could be seen as a kind of style. Gongdiao is actually akin to the key signature in Western music, but it remains unclear how to align each Gongdiao category to a specific key due to historical factors. Therefore, we use Gongdiao as a style label, which incorporates some constraints about the Gongdiao-Style into the model.

4.3.2. The Method to Add Gongdiao-Style Constrain

The Chinese lyrics dataset includes a total of 25 Gongdiao-Styles. Given that having 25 styles for a dataset of about 6,500 songs makes the dataset scarce, with some styles consisting of only a few dozen of instances, incorporating the style constraint directly into the training stage to alter the model parameters might result in an overly strong constraint. Therefore, similar to [**66**], we integrate the Gongdiao-Style constraint into the beam search inference stage and make the strength of this constraint adjustable by hyperparameters. Specifically, we reward generated melody candidates if they satisfy certain Gongdiao-Style constraints, increasing the likelihood to select these rewarded melodies as final outputs. However, different from [**66**], which incorporates reward at each inference step based on local

features, we only apply rewards after completing the final inference step, as the Gongdiao-Style is a global feature of the entire melody sequence.

Before incorporating the constraint, we first train a Gongdiao classifier for melodies. We employ FastText [**34**] as the classifier model, which is a fast, accurate, and widely-used model for text classification, particularly with small datasets. We train this classifier using melodies as inputs and corresponding Gongdiao as labels.

The classifier is applied to the generated candidate melodies as shown in Algorithm 1. For a lyrics sequence from the test set, several melody sequence candidates are generated by the model and we keep the top-n with the highest log-likelihood scores. During the beam search, we maintain $2n$ candidates, to which a Gongdiao-Style award λ is added if the candidate satisfies the Gongdiao constraint. In the end, we retain the top-n candidates with the highest new scores. λ serves as a hyperparameter to control the constraint strength.

Algorithm 1 Adding Gongdiao Constraint in Beam Search

Input: Melody sequence candidates M_{CA_i}, $i \in [1, 2n]$, and M_{CA} is a list including all M_{CA_i}. $score_i$ is the corresponding log-likelihood score for each melody sequence candidate. $real\_style$ is the true Gongdiao of the real melody. $gongdiao\_classifier$ is the Gongdiao classification function.
Output: : Melody candidates $M^{'}_{CA_j}$, $j \in [1, n]$, and $M^{'}_{CA}$ is a list including all M_{CA_j}.
for $i \leftarrow 1$ to $2n$ **do**
    $predict\_style \leftarrow gongdiao\_classifier(M_{CA_i})$
    if $predict\_style = real\_style$ **then**
        $score_i \leftarrow score_i + \lambda$
    end if
end for
$M_{CA} \leftarrow$ sort M_{CA} in descending order by $score_i$
$M^{'}_{CA} \leftarrow M_{CA}[: n]$

Chapter 5

Implementation and Experiments

In this chapter, we provide some details of the implementation and introduce all the experiments we conduct. We first use automatic evaluation to assess global performance. Next, ablation studies will reveal the effect of each added component by automatic evaluation. Then, we introduce the final melody generation evaluation by human listeners. Finally, we show some generation examples.

5.1. Implementation Details

The ratio of the training set, validation set, and testing set is set to 8:1:1 for both the Chinese lyrics dataset and the English lyrics dataset. For the Chinese lyrics dataset which contains two-stage training, we first use all the training data to pretrain the model for 40 epochs, then only use the non-Sanban data in the training set to finetune the model for 10 epochs, which consists of about 2000 data points. We preserve the model with the lowest validation loss in the pretraining stage and use that model for finetuning. After 10 epochs in the finetuning stage, we retain the final model. For the English lyrics dataset which only contains one-stage training, we directly use all the training data to train the whole model and there is no finetune stage. For training in both datasets, the Adam optimizer [**36**] is employed. For the Chinese lyrics dataset, we use a learning rate of 5e-4 in the pertaining stage, while the learning rate in the finetuning stage is 4e-4 with the same optimizer. For the English lyrics dataset, we directly use a learning rate of 5e-4 in the whole training stage. All the models are trained on an NVIDIA A100 GPU or a V100 GPU from Compute Canada [**6**]. Besides, the other model and training configurations are the same as in [**53**], which is also summarized in Table 5.1.

Moreover, during the inference stage, sometimes the generated melody's length is less than twice that of the lyrics, which is not valid because one lyric should correspond to at least one pitch plus one duration. To address this issue, we only allow to generate [*sep*] symbol after the length requirement is met within each sentence.

Name	Value
The number of encoder layers	6
The number of decoder layers	6
The number of attention heads	8
The hidden size of each layer	512
The filter size of each layer	2048

Table 5.1. The hyper-parameters of the model

5.2. Automatic Global Performance Evaluation

Model	Melody Distance
Baseline	2.2965
Our Model	**2.1986**

Table 5.2. Automatic global performance evaluation in Chinese lyrics dataset

We first evaluate our model's global performance through automatic evaluation. Following the method in [**53**], we use the melody duration metric. The computation for the melody duration proceeds as follows: The melody is first transformed into a time series of pitches, with a granularity of a sixteenth note. Subsequently, each pitch in this time series is subtracted by the average pitch of the entire sequence for normalization. Finally, dynamic time warping [**13**] measures the duration between two pitch time series. A lower value in melody duration indicates superior global performance. The experimental results are depicted in Figure 5.2. Here, the SongMASS model serves as the baseline. The comparison shows that our model produces better results than the baseline. We present only the results from the Chinese lyrics dataset. Since we only adapt one module for the model for the English lyrics dataset, its results are covered in the following ablation studies section.

5.3. Ablation Studies

In this section, we introduce the ablation studies we conduct by automatic evaluation, which are used to evaluate the effect of each component. This section includes three subsections: effect analysis of incorporating POS embeddings, effect analysis of incorporating tone embeddings, as well as effect analysis of re-ranking generation candidates based on style constraints.

Model	Validation NLL	Average
(Baseline) Word_{emb}	2.038/2.045/2.012	2.031
Word_{emb}+POS_{emb}	**1.955/2.013/1.978**	**1.982**

Table 5.3. The best validation NLL between baseline and incorporating tone or POS embedding in English lyrics dataset

Model	Validation NLL	Average
(Baseline) Word_{emb}	1.302/1.308/1.293	1.301
Word_{emb}+POS_{emb}	**1.267/1.282/1.265**	**1.271**

Table 5.4. The best validation NLL between baseline and incorporating tone or POS embedding in Chinese lyrics dataset

5.3.1. Effect Analysis of Incorporating POS Embeddings

To evaluate the effects of incorporating POS embeddings, we compare the baseline (the original SongMASS model) with the addition of POS embeddings. Following the objective evaluation method in some music generation work [**33, 32**], we use the Negative Log-Likelihood (NLL) of the lyric-melody validation process to assess the model's fitting degree. NLL is a metric that quantifies the divergence between the generated sequence and the reference sequence. It is computed as follows:

$$NLL = -\sum_{i=1}^{N} log(P(x_i|y_i))$$

In the above equation, x_i is the i-th note in the reference melody sequence, y_i represents the corresponding lyric input, and $P(x_i|y_i)$ denotes the probability that the model predicts the note x_i given the lyric input y_i. The sum runs over all notes in the melody sequence, and N stands for the total number of notes in this sequence. A lower NLL indicates that the model's predicted probability distribution is more closely with the actual outcomes, which implies a better model fitting.

The results in the English lyrics dataset and in the Chinese lyrics dataset are shown in Table 5.3 and Table 5.4, respectively. As shown for experiments in each table, we conducted three repeated experiments using different random seeds to split the dataset. We record the best validation NLL after finishing pretraining and present all the values (separated by symbol /) for the three repeated experiments. The best validation NLL here means we record the best NLL value among all the validation stages after all training epochs. It is observed that in both datasets, Word_{emb} + POS_{emb} performs better than the baseline for all repeated experiments, which illustrates the effectiveness of adding POS embeddings.

5.3.2. Effect Analysis of Incorporating Tone Embeddings

Model	Validation NLL	Average
(Baseline) Word_{emb}	1.302/1.308/1.293	1.301
Word_{emb}+Tone_{emb}	1.263/1.27/1.261	1.265
Word_{emb}+POS_{emb}+Tone_{emb}	**1.255/1.263/1.250**	**1.256**

Table 5.5. The best validation NLL between baseline and incorporating tone or POS embedding in Chinese lyrics dataset

To evaluate the effects of incorporating tone embeddings, similar to the ablation studies conducted in the last subsection, we also compare the baseline (the original SongMASS model) with the addition of tone embeddings. Again, we also use the best validation NLL score to evaluate. Since this adaption is only applied in the Chinese lyrics dataset, this evaluation only contains results in the Chinese lyrics dataset. The results are shown in Table 5.5. We also conduct three repeated experiments using different random seeds to split the dataset. From this table, we can see that both Word_{emb} + Tone_{emb} perform better than the baseline for all repeated experiments. Furthermore, when we combine both POS_{emb} and Tone_{emb} and add them to the original baseline, the performance improves even more for all repeated experiments. This ablation experiment indicates the effectiveness of incorporating tone embeddings into the basic model for the Chinese lyrics dataset.

5.3.3. Effect of Re-ranking Generation Candidates Based on Style

Model	Pitch Distribution Similarity	Average
Baseline	71.43% / 69.97% / 72.12%	71.17%
$\lambda = 0.1$	71.99% / **70.06%** / 72.62%	71.56%
$\lambda = 0.2$	72.03% / 70.04% / **72.89%**	71.65%
$\lambda = 0.3$	**72.04%** / 70.02% / 72.88%	71.65%

Table 5.6. The Pitch Distribution Similarity scores between baseline and adding Gongdiao-Style constraint

We then evaluate the performance of adding the style constraint during the inference stage. This evaluation is also only conducted in the Chinese lyrics dataset. After completing all pretraining and fine-tuning, we run the inference to generate the final melody from the testing data. The precision of the Gongdiao-style classifier we use in Algorithm 1 is about 63.22%. We also present the confusion matrix of this classifier in Figure 5.1. From this confusion matrix, we can see those classes with low precision usually originate from the small data sizes of the corresponding classes.

To evaluate the performance of this component, NLL may not be very suitable since we actually modify the original NLL score. Thus, we use different metrics for evaluation.

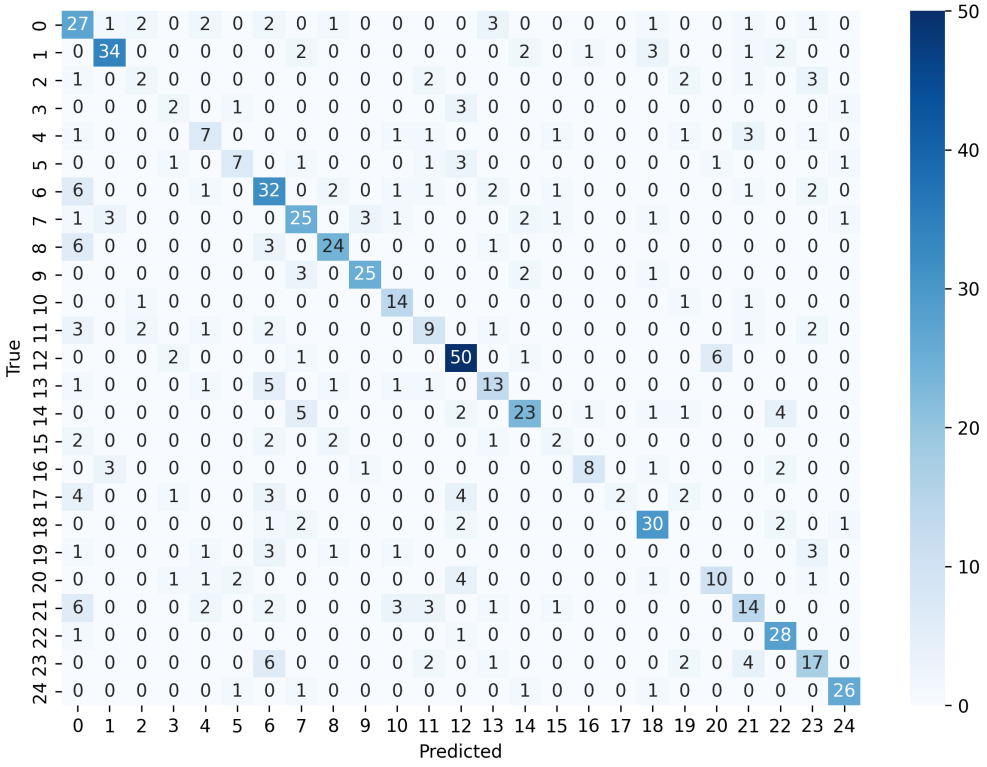


Fig. 5.1. The confusion matrix of the Gongdiao-style classifier

Following the research in [**35**], we use Pitch Distribution Similarity (PD) to evaluate the model. Pitch distribution similarity means the average overlapped area of the pitch frequency histogram distribution between generated melodies and ground truth. The formula for calculating the PD is as follows (from [**53**]):

$$\frac{1}{N_{\mathrm{s}}} \sum_{i=1}^{N_{\mathrm{s}}} OA\left(\mathrm{Dis}_i, \hat{\mathrm{Dis}}_i\right)$$

In the above formula, $OA\left(\mathrm{Dis}_i, \hat{\mathrm{Dis}}_i\right)$ denotes the average overlap between the pitch frequency histogram distribution of the i-th generated melody (Dis_i) and that of its corresponding ground truth. Meanwhile, N_s signifies the total number of songs in the test data set. Better performance is typically associated with higher PD scores. We do not use the Duration Distribution Similarity score, as in [**35**], because Gongdiao-style has little connection with rhythm. Gongdiao-style is similar to the key signature in Western music. Table 5.6 shows the comparison between the baseline (here, meaning Word_{emb} + Tone_{emb} + POS_{emb} from the previous subsection) and adding Gongdiao-Style constraint with hyperparameter value $\lambda = 0.1$ or 0.2 or 0.3. We also conduct three repeated experiments using different random seeds to split the dataset. From this table, it is observed the PD score increases for all three experiments after adding the Gongdiao constraint, which indicates the effectiveness

of this component to some extent. Besides, the increasing degree differs in terms of the hyperparameter values.

5.4. Human Evaluation of the Generated Songs

In addition to automatic evaluation, we proceed with a subjective evaluation by having human listeners assess the overall performance. We only conduct the whole subjective evaluation in the Chinese lyrics dataset since we incorporated the three additional components. We only apply one adaptation in the English lyrics dataset, which may not make enough impact that is perceivable by human subjects. Therefore, we only conduct the subjective evaluation in the Chinese lyrics dataset, but we will show generation examples for both datasets later in the next section. To do the subjective evaluation of Chinese songs, after generating the melodies, we use the Chiyu sound source in synthesizer V [**23**] to synthesize the singing voice. To make the vocals sound less harsh, we lowered the pitch of all notes by one octave. Some demos are presented in `https://singpoem.github.io`. We compare our model with SongMASS (the baseline used in the automatic evaluation) and real music (the ground truth in the dataset). Following the subjective evaluation method in [**33, 61**], we conduct paired comparisons between real music and our model, as well as between our model and SongMASS. We randomly select 10 songs from the original dataset and generate the corresponding melodies using both our model and SongMASS, resulting in a total of 30 songs. We enlisted 6 individuals for our subjective human evaluation. Out of these, 3 have over 10 years of experience or education in traditional Chinese music. The remaining 3, though without formal education in traditional Chinese music, possess experience in other musical areas. During the evaluation, we ask participants the following five questions:

(1) **Overall Musicality:** Which song sounds more musical?
(2) **Overall Style:** Which song sounds more like the traditional Chinese opera style?
(3) **Rhythm:** Which song sounds more natural in terms of the duration of words and pauses?
(4) **Pitch:** Which song has a more natural pitch change or pitch contour?
(5) **Diversity:** Which song sounds more novel and not boring?

We ask participants to answer the above five questions on a Likert-like scale with five options, including "Strong preference for Song 1", "Weak preference for Song 1", "No preference", "Strong preference for Song 2", "Weak preference for Song 2". We also shuffled the order in each pair to avoid the order affecting human evaluation. For each pair, we ask four different people to evaluate, resulting in a total of $20 * 5 * 4$ (song pairs * metrics * people) comparisons. Given that we organized 2 pairs with 4 evaluators each, we obtained 8 evaluations in total. Two participants from the group with more traditional Chinese music expertise were assigned to assess the songs for both pairs.

pairs		metric	wins	ties	losses	p_1 value	p_2 value
real music	our model	overall musicality	18	3	19	0.2888	0.1562
		overall style	13	6	21	0.0390*	0.0407*
		rhythm	14	4	21	0.1915	0.2209
		pitch	17	2	21	0.2587	0.2331
		diversity	10	12	18	0.0351*	0.0396*
our model	SongMASS	overall musicality	15	11	14	0.6745	0.6149
		overall style	8	24	8	1.0	1.0
		rhythm	15	10	15	0.6148	0.4699
		pitch	9	19	12	1.0	0.6890
		diversity	9	25	6	0.0578	0.0232*

Table 5.7. Results of the subjective evaluation. "Win" means the first item in this pair wins, and "Loss" means the second item in this pair wins. The numbers below 'wins', 'ties', and 'losses' represent the total count of participants choosing each option. * denotes there are significant differences (p value < 0.05) between compared pairs under independent two-sample t-test.

The results are detailed in Table 5.7. In this table, the figures under 'wins', 'ties', and 'losses' signify the total number of participants opting for each respective choice. For statistical significance testing, we follow the scoring system from [**33**] where points are assigned as follows: 5 for a strong win, 4 for a weak win, 3 for a tie, 2 for a weak loss, and 1 for a strong loss. Using this system, the p-value is calculated using an independent two-sample t-test, with results displayed beneath the p_1 value. Additionally, considering that participants are likely to select strong options for markedly distinct outcomes, we also use another scoring system: 2 for strong loss, 4 for weak loss, 5 for a tie, 6 for weak win, and 8 for strong win. In this scheme, the disparity between strong and weak options is emphasized more than that between weak options and ties. The p-value for the independent two-sample t-test using this scoring system is listed under the p_2 value.

According to this table, when comparing real music with our model, our model outperforms real music in terms of overall style and diversity, while maintaining similar performance in other metrics. Both p_1 value and p_2 value show statistical differences for these two metrics: overall style and diversity. It is worth noting that due to some missing rhythm information from the original Gongchepu, the real music here is not a perfect rendition by musicians. For instance, Figure 5.5 displays examples of real music, while Figure 5.6 presents music generated by our model. Figure 5.5 represents the Sanban score. Ideally, the Sanban score should be interpreted by musicians, but in our dataset, all notes are assigned the same duration for the Sanban scores. In contrast, Figure 5.6 demonstrates more diverse rhythms. This distinction largely explains our model's superior diversity and style. During the fine-tuning phase, we used only non-Sanban data to train the model, which features more precise rhythms set by musicians. Although the improved performance on some metrics mainly stems from

the imperfect interpretation of the original Gongchepu, it also highlights an advantage of our method. While having musicians interpret the original Gongchepu can be costly, our method provides an automated approach to interpreting the Gongchepu and achieves good results. In addition, comparing our model with SongMASS with the same training process as our model, our model outperforms SongMASS in diversity under p_2 value while keeping the performance in other metrics similar. These observations indicate two things: First, our model is slightly better than the baseline under this subjective evaluation. Second, our model can generate high-quality melodies comparable to real music without the high costs of musicians' interpretations. By contrasting human evaluation with automatic evaluation, we can observe that music evaluation is a difficult task. While automatic evaluation metrics can obtain some measures of comparison with references, they do not necessarily reflect the human perceived performance. Human evaluation should be used as the standard as much as possible. However, human evaluation is not only expensive but also influenced by subjectivity. It is difficult to do it on a large scale. The evaluation of music is an important research topic for future research.

5.5. Melody Generation Examples

In this section, we show some melody generation examples from lyrics. We first present some generation examples from English lyrics, then we present some generation examples from Chinese lyrics.

5.5.1. Melody Generation Examples from English Lyrics

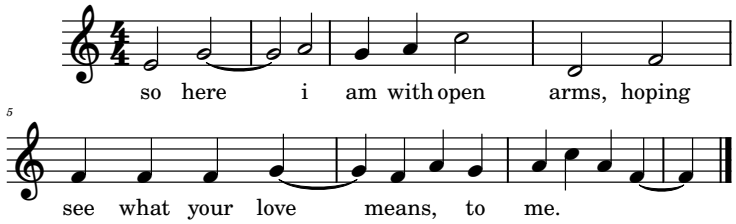


Fig. 5.2. A melody generation example from English lyrics

Here, we show some melody generation examples from English lyrics. Figure 5.2 and Figure 5.3 show two excerpts of generated songs. Comparing these two song examples accompanied by English lyrics to the above song examples with Chinese lyrics, these English songs are less diverse in rhythms. This is mainly because of the types of the original training dataset. For reference, the original song of Figure 5.2 is shown in Figure 5.4. We can see, that the generated song has similar pitches and durations to the reference song (Ignoring the difference in barlines and tuplets since we do not consider these in the training data. We only look at the display of the score).

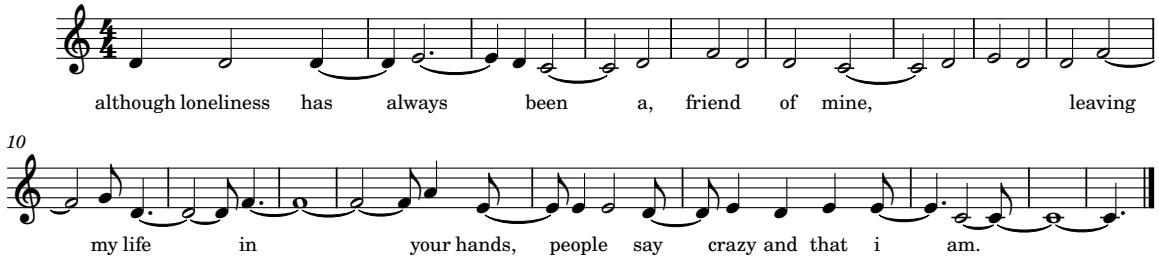


Fig. 5.3. A melody generation example from English lyrics

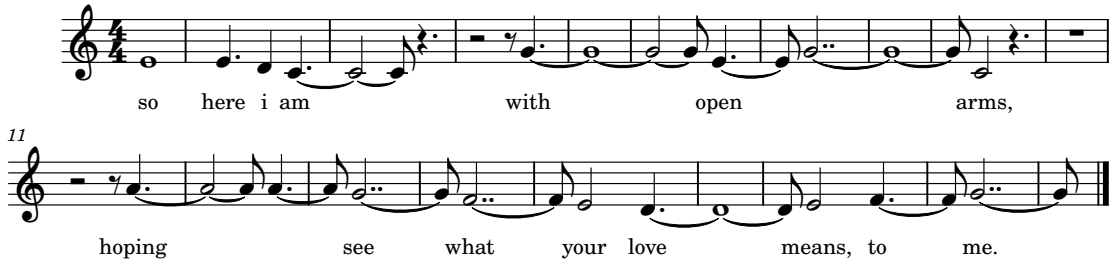


Fig. 5.4. The original song of Figure 5.2

5.5.2. Melody Generation Examples from Chinese Lyrics

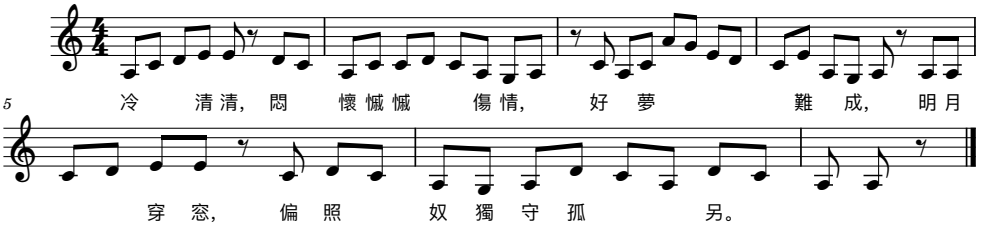


Fig. 5.5. One song from this dataset. The English translation of the lyrics in this song is "Cold, oh so cold; melancholy harbors deep within, hurting the heart; sweet dreams are hard to fulfill. The bright moon pierces through the window; casting its light upon me, solely keeping company with solitude."

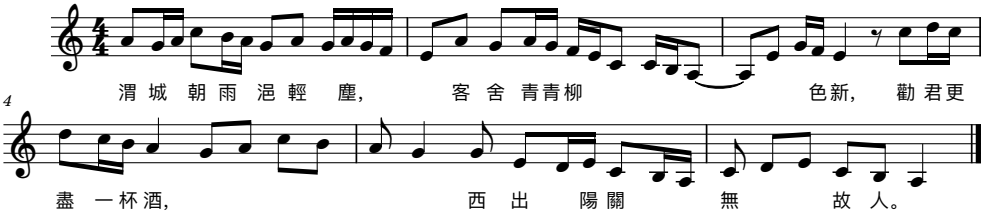


Fig. 5.6. One song generated by our model. The English translation of the lyrics in this song is "Morning rain in Weicheng dampens the light dust. The guest house is lush with the color of new willows. I urge you to finish yet another cup of wine. As west of Yangguan, there will be no old friends."

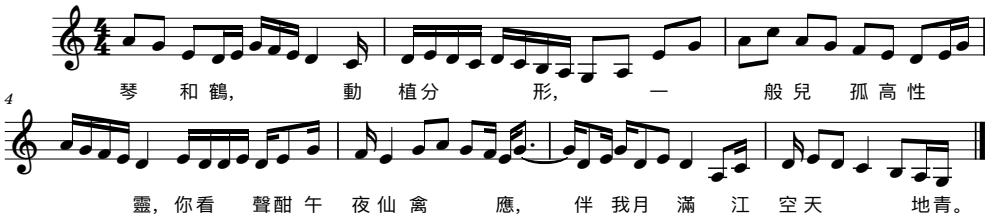
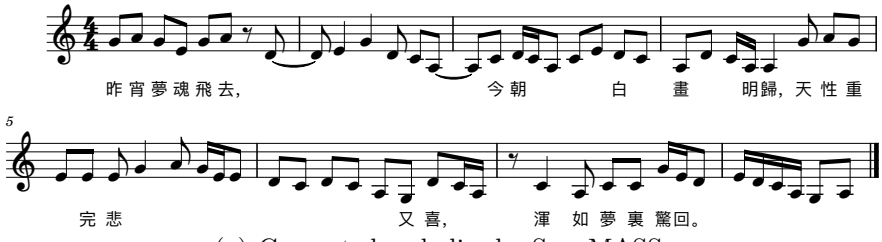
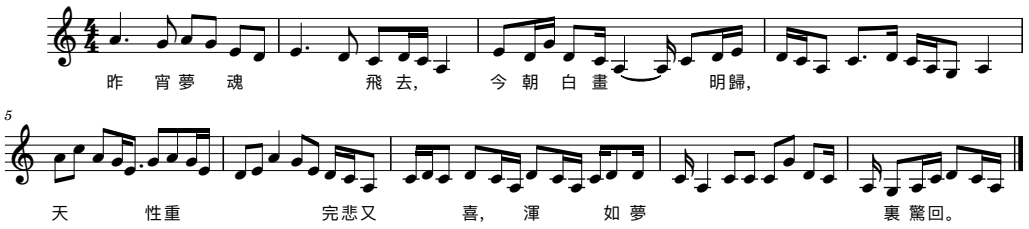


Fig. 5.7. Melody generation example from Chinese lyrics based on our model. The English translation of the lyrics in this song is "The zither and the crane, motion, and stillness take their forms; Such is the lofty and solitary nature of a child. See, in the deep night, the celestial bird responds to the intoxicating sound; Accompanying me, under the full moon, the river, the sky, and the earth are all green."

Then, we present some examples of melody generation from Chinese lyrics. As seen in Figure 5.6 and Figure 5.7, both examples demonstrate a variety of melodies. The note durations in these melodies encompass eighth notes, quarter notes, and rest notes. Moreover, the pitch range is wide, yet it remains within a reasonable range. It is worth noting that the last word in a sentence usually corresponds to a longer duration note or a rest note, signaling the end of a melodic phrase. Besides, Figure 5.8 shows a comparison of the generated melodies between SongMASS and our model. We can see the melodies produced by our model are more diverse in rhythm. Also, from the subjective evaluations, our model can maintain the same high performance in other metrics compared with SongMASS. However, our model does have drawbacks. It tends to generate a longer melody, which might not always sound natural. Future work could consider musical structure factors to alleviate this issue.



(a) Generated melodies by SongMASS



(b) Generated melodies by our model

Fig. 5.8. A comparison of the generated melodies between SongMASS and our model

Chapter 6

Conclusions and Limitations

6.1. Conclusions

This thesis employs deep learning approaches for generating melodies from both English lyrics and classical Chinese poetry. In summary, the contributions of this thesis are as follows:

- We addressed the generation of non-popular music, which has rarely been considered. Specifically, we generate melodies from lyrics in two different kinds of music: one is popular music with English lyrics, and the other is traditional Chinese music with classical Chinese poetry. The former has been extensively researched, while the latter has seldom been explored.
- Furthermore, we alleviate the problem of inadequate modeling of the connection between lyrics and melody in non-popular music. Specifically, we employ deep neural networks to learn from a much larger paired dataset for generating melodies from classical Chinese poetry. This enhances the model's ability to understand the relationship between classical Chinese poetry and its associated melodies. Another motivation behind this endeavor stems from historical context: many classical Chinese poems could be sung in ancient times, but many associated melodies have been lost, leaving only the poetry itself. Given the assumption that the lost melodies share similar elements, such as styles and genres, with the preserved melodies, this thesis employs deep neural networks to model the remaining melodies and their corresponding poems, which may assist in restoring these lost melodies.
- We integrate some music and language knowledge into music generation, which has been done through manual rules in prior research, leading to the limited ability of generalization and adaptability. Our methods allow the model to autonomously encode music theory information for melody generation. Specifically, part-of-speech (POS) embeddings and tone embeddings are incorporated into the model, improving the capture of relationships between prosodic boundaries in lyrics (applicable to both

English and Chinese lyrics) and melody, as well as between the tone of Chinese characters and the pitch of the melody, without manually designed rules.

- We tackle the issue where generated melodies do not encompass stylistic features. Specifically, we integrate style constraints into the inference stage. This adaptation allows the model to grasp the global style features of music to some extent.

After implementing all the above adaptions, we conduct objective evaluations on both datasets and subjective evaluations on the Chinese lyrics dataset. Objective ablation studies demonstrate that all of these adaptations contribute to improving the model's fit to the data. The results of the subjective evaluation reveal that our model can generate high-quality melodies akin to real music.

6.2. Limitations and Future Work

The work presented in this thesis presents several limitations.

- When generating a melody, our model tends to generate a longer melody than the ground truth, which sometimes does not sound very natural.
- In addition, we primarily focused on the score composition level and did not address performance factors. For example, when performing the Gongchepu score, the duration assignment for multiple notes within a beat is typically determined by performers rather than strictly specified in the score.
- Furthermore, we make assumptions about interpreting Sanban data in traditional Chinese music, which may not be realistic. More research on incorporating rhythms into Sanban should be done in the future.
- Besides, we use modern singing voices to sing the melodies for traditional Chinese music; however, it would be better to employ specialized voices with Chinese opera styles and pronunciations, which have not been developed in the literature.

Therefore, substantial work remains to be done.

References

[1] https://www.zebrakeys.com/lessons/preparation/basicmusicnotation/.

[2] https://github.com/yy1lab/Lyrics-Conditioned-Neural-Melody-Generation.

[3] https://www.musicgateway.com\T1\guilsinglrightwhat-is-a-musical-score.

[4] https://colinraffel.com/projects/lmd/.

[5] https://www.reddit.com/r/datasets/.

[6] https://docs.alliancecan.ca.

[7] Jakob Abesser et Gerald Schuller : Instrument-centered music transcription of solo bass guitar recordings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(9):1741–1750, 2017.

[8] Hangbo Bao, Shaohan Huang, Furu Wei, Lei Cui, Yu Wu, Chuanqi Tan, Songhao Piao et Ming Zhou : 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*, pages 499–511. Springer, 2019.

[9] Arnau Baró, Pau Riba, Jorge Calvo-Zaragoza et Alicia Fornés : From optical music recognition to handwritten music recognition: a baseline. *Pattern Recognition Letters*, 123:1–8, 2019.

[10] Leonard E Baum et Ted Petrie : Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[11] Emmanouil Benetos, Simon Dixon, Zhiyao Duan et Sebastian Ewert : Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36(1):20–30, 2018.

[12] Yoshua Bengio, Paolo Frasconi et Patrice Simard : The problem of learning long-term dependencies in recurrent networks. *In IEEE international conference on neural networks*, pages 1183–1188. IEEE, 1993.

[13] Donald J Berndt et James Clifford : Using dynamic time warping to find patterns in time series. *In Proceedings of the 3rd international conference on knowledge discovery and data mining*, pages 359–370, 1994.

[14] Nicolas Boulanger-Lewandowski, Yoshua Bengio et Pascal Vincent : Modeling temporal dependencies in high-dimensional sequences: application to polyphonic music generation and transcription. *In Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1881–1888, 2012.

[15] Francisco J Castellanos, Jorge Calvo-Zaragoza et Jose M Inesta : A neural approach for full-page optical music recognition of mensural documents. *In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[16] Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick et Shlomo Dubnov : Music sketchnet: Controllable music generation via factorized representations of pitch and rhythm. *In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[17] Yu-Hua Chen, Wen-Yi Hsiao, Tsu-Kuang Hsieh, Jyh-Shing Roger Jang et Yi-Hsuan Yang : towards automatic transcription of polyphonic electric guitar music: a new dataset and a multi-loss transformer model. *In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 786–790. IEEE, 2022.

[18] Kyunghyun Cho, B van Merrienboer, Caglar Gulcehre, F Bougares, H Schwenk et Yoshua Bengio : Learning phrase representations using rnn encoder-decoder for statistical machine translation. *In Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[19] Emir Demirel, Sven Ahlbäck et Simon Dixon : Automatic lyrics transcription using dilated convolutional neural networks with self-attention. *In 2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

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

[21] Hao-Wen Dong et Yi-Hsuan Yang : Convolutional generative adversarial networks with binary neurons for polyphonic music generation. *In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[22] Hao-Wen Dong, Cong Zhou, Taylor Berg-Kirkpatrick et Julian McAuley : Deep performer: Score-to-audio music performance synthesis. *In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 951–955. IEEE, 2022.

[23] Dreamtonics : Synthesizer V. `https://dreamtonics.com/en/synthesizerv/`.

[24] Wei Duan, Yi Yu, Xulong Zhang, Suhua Tang, Wei Li et Keizo Oyama : Melody generation from lyrics with local interpretability. *ACM Transactions on Multimedia Computing, Communications and Applications*, 19(3):1–21, 2023.

[25] Editorial Committee of "Anthology of Chinese Traditional Opera Music" : *Anthology of Chinese Traditional Opera Music: Zhejiang Volume (In Chinese)*. 2001.

[26] Yu Gu, Xiang Yin, Yonghui Rao, Yuan Wan, Benlai Tang, Yang Zhang, Jitong Chen, Yuxuan Wang et Zejun Ma : Bytesing: A chinese singing voice synthesis system using duration allocated encoder-decoder acoustic models and wavernn vocoders. *In 2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5. IEEE, 2021.

[27] Siddharth Gururani, Mohit Sharma et Alexander Lerch : An attention mechanism for musical instrument recognition. *In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[28] Donghong Han, Yanru Kong, Jiayi Han et Guoren Wang : A survey of music emotion recognition. *Frontiers of Computer Science*, 16(6):166335, 2022.

[29] Curtis Hawthorne, Ian Simon, Adam Roberts, Neil Zeghidour, Josh Gardner, Ethan Manilow et Jesse Engel : Multi-instrument music synthesis with spectrogram diffusion. *In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2022.

[30] Curtis Hawthorne, Ian Simon, Rigel Swavely, Ethan Manilow et Jesse Engel : Sequence-to-sequence piano transcription with transformers. *In Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[31] Sepp Hochreiter et Jürgen Schmidhuber : Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[32] Cheng-Zhi Anna Huang, Tim Cooijmans, Adam Roberts, Aaron Courville et Douglas Eck : Counterpoint by convolution. *In Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017.

[33] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu et Douglas Eck : Music transformer: Generating music with long-term structure. *In International Conference on Learning Representations (ICLR)*, 2018.

[34] Armand Joulin, Édouard Grave, Piotr Bojanowski et Tomáš Mikolov : Bag of tricks for efficient text classification. *In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, 2017.

[35] Zeqian Ju, Peiling Lu, Xu Tan, Rui Wang, Chen Zhang, Songruoyao Wu, Kejun Zhang, Xiangyang Li, Tao Qin et Tie-Yan Liu : 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 (EMNLP)*, 2022.

[36] Diederick P Kingma et Jimmy Ba : Adam: A method for stochastic optimization. *In International Conference on Learning Representations (ICLR)*, 2015.

[37] Taegyun Kwon, Dasaem Jeong et Juhan Nam : Audio-to-score alignment of piano music using rnn-based automatic music transcription. *In Proceedings of the 14th Sound and Music Computing Conference (SMC)*, 2017.

[38] John D Lafferty, Andrew McCallum et Fernando CN Pereira : Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *In Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, 2001.

[39] Hsin-Pei Lee, Jhih-Sheng Fang et Wei-Yun Ma : icomposer: An automatic songwriting system for chinese popular music. *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 84–88, 2019.

[40] Rongfeng Li, Xinyun Zhang et Minghui Bi : Music poet: A performance-driven composing system. *In Proceedings of the 2016 International Computer Music Conference (ICMC)*, 2016.

[41] Jinglin Liu, Chengxi Li, Yi Ren, Feiyang Chen et Zhou Zhao : Diffsinger: Singing voice synthesis via shallow diffusion mechanism. *In Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11020–11028, 2022.

[42] Peiling Lu, Jie Wu, Jian Luan, Xu Tan et Li Zhou : Xiaoicesing: A high-quality and integrated singing voice synthesis system. 2020.

[43] Ang Lv, Xu Tan, Tao Qin, Tie-Yan Liu et Rui Yan : Re-creation of creations: A new paradigm for lyric-to-melody generation. *arXiv e-prints*, pages arXiv–2208, 2022.

[44] Ethan Manilow, Gordon Wichern, Prem Seetharaman et Jonathan Le Roux : Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity. *In 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 45–49. IEEE, 2019.

[45] Warren S McCulloch et Walter Pitts : A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

[46] Ryo Nishikimi, Eita Nakamura, Satoru Fukayama, Masataka Goto et Kazuyoshi Yoshii : Automatic singing transcription based on encoder-decoder recurrent neural networks with a weakly-supervised attention mechanism. *In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 161–165. IEEE, 2019.

[47] Expecto Patronum : What a hard journey, 2022. Aisong Contest 2022 Participants.

[48] Nikki Pelchat et Craig M Gelowitz : Neural network music genre classification. *Canadian Journal of Electrical and Computer Engineering*, 43(3):170–173, 2020.

[49] Yi Ren, Xu Tan, Tao Qin, Jian Luan, Zhou Zhao et Tie-Yan Liu : Deepsinger: Singing voice synthesis with data mined from the web. *In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1979–1989, 2020.

[50] Yi Ren, Xu Tan, Tao Qin, Jian Luan, Zhou Zhao et Tie-Yan Liu : Deepsinger: Singing voice synthesis with data mined from the web. *In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1979–1989, 2020.

[51] David E Rumelhart, Geoffrey E Hinton et Ronald J Williams : Learning internal representations by error propagation. Rapport technique, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[52] Casey Schoenberger : Nine modes manual online. `https://wapp.lib.polyu.edu.hk/ninemodes`.

[53] Zhonghao Sheng, Kaitao Song, Xu Tan, Yi Ren, Wei Ye, Shikun Zhang et Tao Qin : Songmass: Automatic song writing with pre-training and alignment constraint. *In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

[54] Arun Solanki et Sachin Pandey : Music instrument recognition using deep convolutional neural networks. *International Journal of Information Technology*, 14(3):1659–1668, 2022.

[55] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu et Tie-Yan Liu : Mass: Masked sequence to sequence pre-training for language generation. *In International Conference on Machine Learning*, pages 5926–5936. PMLR, 2019.

[56] Fabian-Robert Stöter, Stefan Uhlich, Antoine Liutkus et Yuki Mitsufuji : Open-unmix-a reference implementation for music source separation. *Journal of Open Source Software*, 4(41):1667, 2019.

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

[58] Bryan Wang et Yi-Hsuan Yang : Performancenet: Score-to-audio music generation with multi-band convolutional residual network. *In Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1174–1181, 2019.

[59] Paul J Werbos : Generalization of backpropagation with application to a recurrent gas market model. *Neural networks*, 1(4):339–356, 1988.

[60] Yusong Wu, Ethan Manilow, Yi Deng, Rigel Swavely, Kyle Kastner, Tim Cooijmans, Aaron Courville, Cheng-Zhi Anna Huang et Jesse Engel : Midi-ddsp: Detailed control of musical performance via hierarchical modeling. *In International Conference on Learning Representations*.

[61] Yusong Wu, Ethan Manilow, Yi Deng, Rigel Swavely, Kyle Kastner, Tim Cooijmans, Aaron Courville, Cheng-Zhi Anna Huang et Jesse Engel : Midi-ddsp: Detailed control of musical performance via hierarchical modeling. *In International Conference on Learning Representations (ICLR)*, 2022.

[62] LiLi Xu : Exploring the translation of kunqu gongchepu: Formulas and concise translation methods (In Chinese). *Journal of the Central Conservatory of Music*, (3):70–84, 2016.

[63] Yinliu YANG : *An Introduction to Gongchepu (In Chinese)*. 1962.

[64] Yi YU, Abhishek SRIVASTAVA et Simon CANALES : Conditional lstm-gan for melody generation from lyrics. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 17(1):1–20, 2021.

[65] Lu YUN : *Jiugong Da Cheng Nanbei Ci Gongpu (In Chinese)*. 1746.

[66] Chen ZHANG, Luchin CHANG, Songruoyao WU, Xu TAN, Tao QIN, Tie-Yan LIU et Kejun ZHANG : Relyme: Improving lyric-to-melody generation by incorporating lyric-melody relationships. *In Proceedings of the 30th ACM International Conference on Multimedia (MM)*, 2022.