國 立 清 華 大 學

碩 士 論 文

一個基於卷積的高效自注意力旋律生成神經網絡模型

An efficient music generator based on CNN with Attention mechanism

系所別：

計算與建模科學研究所

學號姓名：

108026466 劉真濡 Zhen-Ru Liu

指導教授：

蘇豐文 Soo, Von-Wun 博士;陳人豪 Chen, Jen-Hao 博士

中 華 民 國 一 百 一 拾 年 七 月

摘要

我們設計了一個基於卷積與自註意力機制結合而成的神經網絡模型用於單音音樂旋律生成。採用 WAV 格式文件作為資料集而並非使用更廣泛的 MIDI 格式，提取主旋律音高，持續時間的序列作為數據集，進行訓練。得到了一個易於訓練，易部署的旋律生成模型。最後，我們引入一套基於統計的指標進行模型生成樣本的評估。

關鍵字: 深度學習，自註意機制，音樂生成。

Abstract

An efficient music generator based on CNN based Attention Neuron Network model

Zhen-Ru Liu Advisor: Dr. Jen-Hao Chen; Dr. Von-Wun Soo

Institute of Computational and Modeling Science,

National Tsing Hua University, Hsin-Chu, Taiwan

We designed a neural network model built on the combination of convolution and self-attention mechanism for monophonic music melody generation. The WAV format file is utilized instead of the more widely utilized MIDI format, and the sequence of the main melody pitch and duration is extracted as a dataset for training. A melody generation model that can be trained easily and deploy is obtained. Finally, we lead a set of statistics-based indicators into evaluateing the model generation samples.

Keywords: Deep Learning, Self-Attention, Music Generation.

Acknowledgements

I am very grateful to Professor Dr.Von-Wun Soo and Associate Professor Dr.Jen-Hao Chen for their help and guidance in completing my thesis, reading my paper and helping me fix all the mistake i have made in my paper. And I also appreciate those researchers who proposed those previous works, inspired my research idea. Thanks to all those who participated in the paper questionnaire survey. Without their help, this would be impossible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Artificial Intelligence (AI) is the most heated fields. The most exciting part is involve AI in emotional and creative artistic work: drawing, writing novels, and even creating music. But, creating music is more difficult than the two previous tasks. The data processing of music is the first challenge. Images and text samples can be directly mapped to corresponding pixel matrix and word vectors, but music can't. Music is an emotional language that is highly organized and abstract but lacking actually meaning. That's the reason why music is difficult to process [2]. On the other hand, music performances are full of interaction and flexible changes with composition, making music difficult to describe.

The above reasons make the difficulties of automatic music generation using the AI designation. However, in recent years, we have continuously observed excellent cases of music AI architectures. Those AIs can be divided into two parts based on the data forms [3]: symbolic AI (will be called symbolic forms below) and sub-symbolic AI (will be called wave forms below). Symbolic forms mean using files like MIDI, XML, and other formats as datasets to train AI. Those records of performance representations information the stream annotations, pitches, beats, chords in the files, can be directly regarded as training datasets. Wave forms mean using files that record sound waves such as the WAV format to training AI, which need signal processing as data pre-processing.

For convenience, symbolic forms become the best choice of most researchers in the music generation field. Wave forms need to spend extra time to design the signal processing SOP to

process the files and extracting the data set for the training. From 2016 to now, there are only 14 AI belongs to wave forms deal with music generation, indicate this kind of AI is rare [4–18].

For AI structures, any well-known structure has good results in music generation. First version of magenta [19], SampleRNN [5], and a deep LSTM model [11] are the researches based on the RNN family (Recurrent Neural Network [20], LSTM (Long Short Term Memory) [21]) for music generation. MidiNet [22], MuseGAN [23] and GANSynth [9] are generation AI based on Generative Adversarial Networks (GAN) [24].

COCONET [25] has been proposed by Huang et al, which proved the Convolution Neural Network (CNN) [26] also has the ability to handle music. A hierarchical latent vector model was proposed for music generation [27] based on Variational Autoencoder (VAE) [28] by Google. Started in 2019, the Transformer [1] models were deployed to music generation. Those models were originally for text generation and translation. There are two example models: the Music Transformer [29], this model is also the core of the new version of Magenta, and the Pop Music Transformer [30], which focused on pop music generation.

In this paper, we combine CNN [26] and mainly two kinds of Attention models, Addictive Attention [31] and Scaled Dot-Product Attention [1] to propose a new monophonic melody generation waveform AI. Among them, we change Attention structure to make its' training more efficient by adding the idea based on the work of Li et al. [32] and the residual network (ResNet) [33]. Our model requires fewer training parameters and occupies less memory when it is trained.

Chapter 2

Backgrounds

In this chapter, some in need background acknowledge are introduced: The basic music theories, digital signal processing, and deep learning models.

2.1 Music

In this section, we will introduces some of the fundermental music theories and knowledge. The references come from the entrys of Grove Music Online, which is the authoritative online dictionary in the field of music.

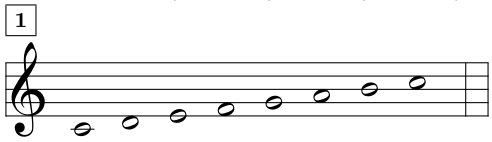
2.1.1 Note

In music, a note, is the smallest unit, which is also a symbol denoting a musical sound [34].

2.1.2 Scale

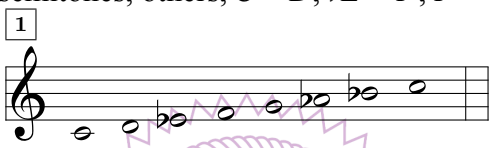
A series of single notes progressing up or down stepwise [35]. Each different scale has different numbers of notes and the arrangement of different intervals between notes. Among them, a scale also means a sequence long enough to define unambiguously a mode, tonality, or some special linear construction. Thus, those notes within an octave, and in a scale are used as the basis of fundamental material for composing. Scales are arbitrary, the number of different scales that can be formulated is theoretically nearly infinite. For example, the most common

scales are major (alias Ionian) and minor (alias Aeolian). Major and minor both has 8 key notes in an octave (see 1. 1. 2, Octave), with 5 whole tone [36], and 2 semitone [37]. thus, the two scales are also called heptatonic scale [38], or heptachord [39]. The note of the major scale is arranged as $C-D-E-F-G-A-B-C$, and the note C is the tonic [40] of major, in a word, a tonic means the cornerstone tone of a scale. $E-F$ and $A-B$ are the two semitones, other 5 paired adjacent notes to each other:$C-D$, $D-E$, $F-G$, $G-A$, are whole tones.



scale C:maj (major with tonic C)

Arrangements of minor is: $C-D-\flat E-F-G-\flat A-\flat B$, and the note A is the tonic. . $D-\flat E$, $G-\flat A$ are the two semitones, others, $C-D$, $\flat E-F$, $F-G$, $\flat A-\flat B$ are whole tone.



scale C:minor (minor with tonic C)

2.1.3 Octave

Interval of 12 semitones, counting bottom and top notes. Notes an octave apart have same letternames.

2.1.4 Motive/Motif

It is a short musical idea, melodic, harmonic, rhythmic, even any combination of these music elements could be a motif [41]. A motif decides the music ”figure” and ”fundamental” [42], which also is the smallest ”meaningful” unit in a piece of music and could be any size. A motif also promotes the evolution and development of music melody. It plays a big role in a piece of music. In a complex music work, there usually is more than one motif.

2.1.5 Melody

Melody [43], defined as play a music with a specified scale in musical time in accordance with given cultural conventions and constraints .

2.1.6 Counterpoint

A term describes the combination of simultaneously sounding musical lines according to a system of rules. In other words, it's a set of rules and regulations that define the timing of notes usage and arrangements for those mutiple musical lines in a piece of music [44].

2.1.7 Five core elements of music

1. The scale (see Chapter 2.1.2), the backbone of music, determines a piece of music.
2. Scale and the beats decide the basic style of music. It's the melody.
3. Keyboard chord and other voice type are added to accompany the main melody, that describes the feature of music, following the rules of the counterpoint (see Chapter 2.1.6).
4. Instruments. Appropriate instrument performance enhances the composer's auditory experience of emotional expression.
5. Music form. Through continuous practice, music form defines the musical works constitute a whole and the structural rules of each part, as well as a systematic explanation of the structural form, the combination of the theme and non-theme components, and the tonality layout of the musical works.

2.2 Degital Signal Processing

In this section we introduce the basic knowledge of digital signal processing, Mainly talk about Fouier transform, and convolution. Convolution is also related to the Convolution neural network(CNN).

2.2.1 System

A system transforms the signal x(t) and delivers a signal y(t), the result of this alteration. A bunch of next concepts, knowledge we are talking about are based on the concept of system. In other words, anything that can generate another signal after input one signal is called a system.

2.2.2 Convolution

Convolution is an important operation in analytical mathematics, often used in signal processing and image processing [45–47]. A convolution operator can be in one-dimensional [46, 47], and in two-dimensional [48]. The relationship between the input to a linear shift-invariant system, x(n), and the output, y(n), is given by the convolution sum.

One-Dimensional Convolution

One-dimensional convolution is typically used in signal processing to calculate the delay accumulation of the signal. Suppose that a signal generator generates a signal x_t at every time t, and the attenuation rate of its information is w_k, that is, after k − 1 time steps, the information is w_k times the original. At time t, the received signal y_t is the superposition of the information generated at the current time and the delayed information at the previous time,

$$y_t = \sum_{k=1}^{m} w_k \cdot x_{t-k+1}. \tag{2.1}$$

those $w_1, w_2, w_3, \cdots, w_k$ are called filter or convolution kernel, where in (2.1), denots the length of filters. The formula describes a of those filters $w_1, w_2, w_3, \cdots, w_k$, convolution with a sequence of signal : $x_1, x_2, x_3, \cdots, x_{t-k+1}$. In a signal system, an output sequence {y(n)} of discrete-time linear filter also describe as follow,

$$y(n) = (x \star h)(n) = \sum_{k=-\infty}^{+\infty} x(k)h(n-k) = \sum_{k=-\infty}^{+\infty} h(k)x(n-k) \tag{2.2}$$

where the sequense {h(n)} is called the impulse response, characterizes the filter, and $x \star h$ denotes the convolution operator see Eq. (2.2) [47].

Two-Dimensional Convolution

For image processing, we extend one-dimensional convolution to two-dimensional, 2D convolution is the name of the operation that associates the two 2D matrix, $X \in \mathbb{R}^{M*N}$ denotes a image and $X \in \mathbb{W}^{m*n}$ denotes the filter, in general, $m \ll M, n \ll N$, with a matrix output y_{ij}

of convolution.

$$y_{ij} = W \star X = \sum_{u=1}^{m} \sum_{v=1}^{n} w_{uv} \cdot x_{i-u+1,j-v+1} \tag{2.3}$$

2.2.3 Fouier transform

The Fourier representation provides a way to map the signal to another ”domain” for operation, it plays an extremely important role in both continuous-time and discrete-time signal processing [49]. In addition, the Fourier transform provides a different way of interpreting signals and systems. The spectral content X(f) of the function $x(t) \in L_1(\mathbb{R}) \cap L_2(\mathbb{R})$ can be expressed by integration using a complex exponent [50]. We call this integral Fourier transform,

$$X(f) = \int_{\mathbb{R}} x(t) e^{-2j\pi ft} dt \quad \longleftrightarrow \quad x(t) = \int_{\mathbb{R}} X(f) e^{2j\pi ft} df \tag{2.4}$$

where $|X(f)|$ is called spectrum of $x(t)$. What makes Fourier representations particularly useful is the mapping of convolution operations to multiplicative properties [49], as Eq. (2.4) shows. For example, that means, if we convolution a filter with a signal in time-domain, we can get a product of every componets in frequency-domain, so that's also a magnificient tool for signal processing of music.

2.2.4 Spectral representation

The main goal in the spectral study of a signal is to find out how to decompose this signal as a sum of sines. By using the sampling theorem, a continuous-time signal can be converted into a series of values in the frequency band (- Fs/2, Fs/2), obtained by using a filter with a gain equal to 1, and then the frequency Fs. Then, we need to get the frequency response of the target system by the idea would be to calculated,

$$Q(f_0) = \sum_{n \in Z} x(n) e^{(-2\pi f_0 n)} \tag{2.5}$$

which can measures how similar the sequences {x(n)} and $\{e^{(-2\pi f_0 n)}\}$ are. In other word, evaluate every f_0 of the every component in signal x(n), what exactly Discrete-Time Fourier

transform does [49, 51].

2.3 Deep Learning Models

In this section we introduce the deep learning neural networks and algorithms are invovled in this research.

2.3.1 Convolutional Neural Network (CNN)

We discuss the convolution before (see Chapter 2.2.2). For our research, because we deal with time-series data(music), we only discuss 1-D convolution. Convolutional Neural Networks are generally composed of convolutional layers, pooling layers, and fully connected layers [26]:

I. Convolutional Layers: Convolutional neural network (CNN) is a neural network that uses convolution operation as one of its layers, The function of the layer is to extract the features of a local area. Different convolution kernels are equivalent to different features extractors [26, 52, 53].A CNN has more than one convolution layers, the lth net input (Net activity value that has not passed through the nonlinear activation function) z^l, is the convolution of the activitied values $a^{(l-1)}$ of the l-1th layer and the filter $w^{(l)} \in \mathbb{R}^M$,

$$z^{(l)} = w^{(l)} \otimes a^{(l-1)} + b^{(l)} \tag{2.6}$$

the filter w^l is the weight matrix that to be trained, $b^{(l)} \in \mathbb{R}$ is the bias to be tained.The convolution layer has two extremely important properties: 1. Locally Connection: Each neuron in the lth convolutional layer is only connected to the neurons in local windows of the l-1th layer are connected to form a local connection network. The number of connections between the accumulation layer is $n^{(l)} * m$. m is the filter size; 2.Shared Weight: By Eq.(2.6), in fact, $w^{(l)}$ as hyperparameters are the same for all neurons in the lth layer.

II. Pooling Layer: Also called subsampling layer, pooling layer it plays a role in performing feature selection, reduce the number of features, thereby reducing the number of parameters, and avoid overfitting to some degree [26, 52].There three mainly pooling function :

1. Maximum Pooling: Generally, the maximum value of all neurons in a region is taken.

$$Y_{m,n}^{d} = \max_{i \in R_{m,n}^{d}} x_i \tag{2.7}$$

Where x_i is the activited value of every neurons in sampling area R_k^d.

2. Mean Pooling: Generally, the average value of all neurons in a region is taken.

$$Y_{m,n}^{d} = \frac{1}{\left|R_{m,n}^{d}\right|} \sum_{i \in R_{m,n}^{d}} x_i \tag{2.8}$$

Subsampling the $M \times N$ regions of each input feature map X^d, for every input feature map X^d(feature tensor of raw inmage abstracted by convolution layers), then the output is $Y^d = \{Y_{m,n}^d\}, 1 \leq m \leq M, 1 \leq n \leq N$.

3. Global Pooling: Global Pooling [54] reduces the dimensionality from 3D to 1D, can be the maximum or the average or whatever other pooling operation. For example, a tensor shape(N, W, H, C), where n is the number of feature maps, w, h, and c is the width, and height, and channels of each feature map. Global pooling returns a tensor shape (N, 1, 1, C). That means multiplying the weight by $\frac{1}{W*H}, \forall c_i \in C$.

III. fully connected layers: The fully connected layers are similar to the neurons arranged in traditional forms of ANN. This layer usually is put at the last in a CNN, attempt to produce class scores from the activations, to be used for classification [26].

2.3.2 Residual Network

Residual network (ResNet) improves the efficiency of information dissemination, by adding Shortcut to the nonlinear convolutional layer (the idea is not limited to convolutional neural networks) [33].

We expect a non-linear unit (which can be one-layer or multi-layer Convolutional layer or other layer like recurrent neural network (RNN)$\cdots$) in a deep neural network $f(x;\theta)$ to approximate an target function as h(x). h(x) can be devided into identity function and residue function,

$$h(x) = \underbrace{x}_{\text{Identity Function}} + \underbrace{(h(x)-x)}_{\text{Residue Function}} \tag{2.9}$$

Actually, the residual function is more easier to learn than h(x). Therefore, the original optimization problem can be transformed into: let the nonlinear unit $f(x;\theta)$ approximate the residual function h(x)-x, and use $f(x;\theta)$+x to approach h(x).

2.3.3 Long Short-Term Memory

Long Short-Term Memory (LSTM) first published in 1997 [21], is the one of most famous and successful deep-learning architecture, enhenced RNN, to solve the gradient explosion or disappearance problem [55, 56] of the RNN [57], has a better memory capacity. We first discuss RNN and it's drawback.

In order to process those data to be predicted, which has an important relationship (text sentence, audio, and other time-series data), we need deep learning models that have the recalling ability. The recurrent neural network was born for this. There is a recurring relationship between its neurons, in the same hidden layer. The first recurrent network is called the Simple Recurrent Network [57], has only one hidden layer, originally for dealing with text learning and understanding in natural language processing. Suppose that at time t, the input to the network is xt, and the hidden layer state (ie, the hidden layer neuron activity value) H_t is related not only to the input x_t at the current moment but also to the hidden layer state h_{t-1} at the previous moment,

$$\begin{aligned} z_t &= Uh_{t-1} + Wx_t + b \\ h_t &= f(z_t) \end{aligned} \tag{2.10}$$

where z_t is the net input (see Chapter 2.3.1), $f(\cdot)$ is a nonlinear activation function, normally is a Logistic function ($\frac{1}{1+e^{-t}}$) or a Tanh ($\frac{e^t-e^{-t}}{e^t+e^{-t}}$) function, U denotes the weights matrix of state-state, and W is weights matrix of the state-input.

When the sequence becomes longer and longer, gradient explosion or disappearance problem [55, 56] occurs, leading to a difficulty in modeling the relationship between the states in

a long range. That is called long-term dependencies problem. There are two main reasons: Firstly, it is related to the backpropagation through time algorithm, used for calculating the gradients of loss function of a hidden state in a Recurrent Neural Network (RNN) [58]. Secondly, is the memory capacity problem: As h_t keeps accumulating and storing new input information, saturation will occur. In other words, the information that can be stored in the hidden state h_t is limited. As more and more contents are stored in the memory unit, more and more information is lost. For more mathematic details and information about the problem gradient explosion or disappearance problem, see [55, 56, 59], which we do not discuss here.

Another drawback is unable to parallel processing, for it's time order. The information i_t at time t cannot be processed before the information i_{t-1} at time t-1 is completely processed by the neuron.

One of the first-class solutions of the long-term dependencies problem is adding the gating mechanism to control the accumulation speed of information, including selectively adding new information, and selectively forgotten previously accumulated information. Then those recurrent neural networks are called the Gated Recurrent Neural Network (Gated RNN) [21, 60]. LSTM is a class of Gated RNN. But it is also cannot process parallized.

LSTM [21] enhaced the RNN dealing with long-term dependencies by adding the idea of internal state and gating mechanism.

Internal state: The internal state c_t is added specialized for linear recurrence information transmission and at the same time (non-linear) output information to the external state h_t of the hidden layer,

$$\begin{aligned} c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \tag{2.11}$$

where $\odot$ is vector element product, c_{t-1} is memory unit at the last moment t-1; $\tilde{c}$ is the candidate state c_t for updating the state(see Eq. (2.11)), where:

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{2.12}$$

At each time t, the internal state c_t of the LSTM network records the historical information

up to the current time f_t, i_t and o_t are the forget gate, input gate and output gate control the transmission path for information:

Forget Gate: Forget Gate f_t controls the internal state of the last moment c_{t-1} how mch information need to be forgotten, where:

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{2.13}$$

Input Gate: Input Gate i_t controls the current state of the candidate state $\tilde{c}_t$ how much information needs to be saved, where:

$$i_t = \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \tag{2.14}$$

Output Gate: Output gate o_t controls how much information of the internal state c_t at current time t should be output for updating external state h_t,

$$o_t = \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \tag{2.15}$$

where $\sigma(\cdot)$ is the Logistic function, with output range (0, 1), x_t is input of current time t, h_{t-1} is the external state of last time t-1 (Eq. (2.11) also indicates that f_t, i_t are control the updating of internal state together, external state updating onlt related to output gate.).

2.3.4 Attention

The amount of information that can be stored in a neural network is called network capacity, which is proportional to the number of neurons and the complexity of the network. The more information to be store, the larger neurons are needed or the network must be more complex, which causes the parameters of the neural network to increase exponentially. Network Capacity also occurs in biological neural networks, such as those in our human brain.

There is a lot of external input information received at each moment. However, with limited resources, the human brain cannot process excessive input that information from sight, hearing, and other feelings at the same time. One of the most important solutions to information overload

is the attention mechanism, as a resource allocation scheme, allocates computing resources to more important tasks. First applied in machine translation, using the attention model to improve recurrent neural networks Long-term dependence.

The attention in cognitive neurology and in deep learning is different to some degrees. We only talk about the attention mechanism in deep learning here. There are two modes of attention in our brains: Selective Attention and Saliency Based Attention. For more information about attention in cognitive neurology, we provide a reference here, see Micko, 1984 [61].

Attention in Deep Learning

For selecting information related to some particular part, from a sequence of input vector $[x_1, \cdots, x_n]$, an expression q is required, which is called query vector, then through a scoring function to calculate the correlation between each input vector and the query vector.

We use the attention variable $z \in [1, N]$ to represent the query vector q can be the index position of dynamically generated selection information. That is, in Eq. 2.16a means the probability that the ith input vector is selected when the q is given. In order to facilitate calculation, we adopt a "soft" information selection mechanism [31]. First calculate probability of the i-th input direction quantity α_i to be choosen, given by q and X,

$$\alpha_i = p(z = i \mid X, q) \tag{2.16a}$$

$$= \text{softmax}\left(s\left(x_i, q\right)\right) \tag{2.16b}$$

$$= \frac{\exp\left(s\left(x_i, q\right)\right)}{\sum_{j=1}^{N} \exp\left(s\left(x_j, q\right)\right)} \tag{2.16c}$$

Among them α_i is called attention distribution, s(x_i , q) is the attention scoring function, can mainly use the following methods to calculate,

1. Addictive Attention (Bahdanau et al, 2014) [31] $s\left(x_i, q\right) = v^{\mathrm{T}} \tanh\left(W x_i + U q\right)$, (2.17)
2. Dot-Product Attention (Luong et al, 2015) [62] $s\left(x_i, q\right) = x_i^{\mathrm{T}} q$, (2.18)
3. Scaled Dot-Product (vaswani et al, 2017) [1] $s\left(x_i, q\right) = \frac{x_i^{\mathrm{T}} q}{\sqrt{d}}$, (2.19)

More generally, we can use key-value pair format to represent input information, where "Key" is used to calculate the attention distribution α_n, "value" is used to calculate aggregate information. $(K,V) = [(k_1,v_1),\cdots,(k_N,v_N)]$ denotes n groups input information, the qurey vector q is given, the attention function is,

$$\text{att}((K,V),q) = \sum_{n=1}^{N} \alpha_n v_n \tag{2.20a}$$

$$= \sum_{n=1}^{N} \frac{\exp\left(s\left(k_n,q\right)\right)}{\sum_j \exp\left(s\left(k_j,q\right)\right)} v_n, \tag{2.20b}$$

$s(k_n,q)$ in Eq.2.20b is the scoring function. This actually is the Query-Key-Value mode [1]. We will discuss more about Attention models later.

2.3.5 Time-Distributed Operator Layer

The last topic in this chapter is the time-distributed layer, which is a wrapper class in Tensorflow and keras [63, 64]. This wrapper allows to apply a layer to every temporal slice of an input. Consider the input is a tensor with shape (n, i, j), where n is the number of matrix samples, where each sample has a size of $i * j$, i vectors composed of j dimensions, also can consider the matrix as a time series, with i steps, each steps has a data sequence length is j. Time-Distributed layer is to apply a neural layer (Fully connected layer, CNN) to those ith specific vectors, giving the model a one-to-many, many-to-many ability, increasing the dimensionality and comlexity of the model.

Chapter 3

Experimental Details

3.1 Dataset

The dataset we use here is not opensourced. We have purchased High-Resolution format classical music audio files (means the sampling frequencies equivelent or larger than 96KHz, in DSD, DSF, FLAC) from the Sony Music, BIS Records, Deutsche Grammophon Gesellschaft, and other music record companies. The number of music is about 3000.

3.1.1 Reasons for choosing the dataset

The reason for choosing this data set is as follows, all of which is to make it easier to achieve the goal that to train the model to compose some new melodies:

I. Quality:

1. Quality Assurance of the sound: The High-Resolution format audio files of these record companies will avoid noise during recording due to their high sound quality requirements. Therefore, there is no need to use signal processing and filtering to filter out noise.

2. Quality Assurance of the music: Although there some free sample on the internet, we have no way to know who completed the compilation of those free music files on the Internet, and it is usually not the work of those music masters. Purchasing sample can guarantee

music arrangement and getting more professional support of music theory.

II. Convenience:

1. Convenience for data abstracting: In addition to ignoring noise in a specified audio format, the special properties of classical music can also help simplify data abstraction. In the performance of classical music, multiple parts are playing different instruments (symphony, concerto) with simultaneously sounding musical lines at the same time. Even solo, there is the concept of parts, such as accompaniment. It is also called the main part, and the core melody line of its performance represents the whole piece of music. Therefore, the first attribute is here. In classical music, the dominant/main part always has the largest volume (amplitude). We can use this attribute to get the main melody line. Classical music has another property that is compared with other modern music genres, it follows the rules of music theory more, especially counterpoint (see Chapter 2.1.6), which can reduce the probability of errors in the extraction of the main melody, for example, mistakenly counting the note components belongs to other musical lines as the main melody.

3.1.2 Feature extraction and processing of the dataset

Feature extraction

Now it goes to the feature extraction by data processing. The most significant for music modeling is the pitch (note frequency) and beat (time duration of the note). Thanks to the properties of high-resolution format audio files and classical music, we can get results with a simple process:

1. Translation: First, we need to get the file encoding of DSD and DSF, and then we must translate these files into WAV format because their encoding cannot be processed directly.

2. Tracing the main melody line: Using signal processing, we cut the time axis, take 0.025s, as a segment, and grab the frequency series corresponding to the largest amplitude. The time length is tested, duration interval between 0.025s-0.04s can get the frequency series closed to the raw musical lines.

3. Onset detection: At the last step we have got the frequency series, in this step we focus on how to get the note beats. Using the onset detection, we can get all the note beats.

Processing the 3 step to each High-Resolution format then we can get the whole frequency and beat series of all music. The raw signal processing tool is provided by Downey, 2016 [45]. The onset detection function is called librosa.onset.onset_detect, provided by librosa [65], a professional music signal processing toolbox.

3.1.3 Dataset processing

We still need to process the frequency and the beat series what we have got.

Frequency series: For frequency series, there are some errors to the standard frequencies of note pitches, if it is directly used as data, it will cause out of tune. To fix this, we need to build a dictionary of standard music note pitches (frequency values). There is a total of 120 different keys of pitch, in ten octaves, each octave has 12 keys. Start at C0 (note key C in the first octave) with a pitch frequency is about 16.325Hz. Then use $16.352 * 2^{\frac{g}{12}}, \forall g = 0, 1, \cdots, 120$ to get all frequencies of every pitch. Then we build intervals $(16.352 * 2^{(\frac{g}{12} - \frac{1}{24})}, 16.352 * 2^{(\frac{g}{12} + \frac{1}{24})}), \forall g = 0, 1, \cdots, 120$ for every standard pitch, which are those intervals center. If the value in the frequency series is not the standard pitch value, check which interval this value falls in, then replace the value as the standard pitch frequency value, $16.352 * 2^{\frac{g}{12}}$.

Beats series: For time duration series, every value needs to normalize as standard note beats. It needs to count the tempo (velocity), measured in bpm (beats per minute), confirmed by calculating how many quarter notes per minute the music melody has. Librosa [65] also provides the function to count bpm, is called librosa.beat.tempo. Bpm is defined by how many quarter notes (or equivalently) plays in 1 minute. After we get bpm, we know the number of quarter notes and can get the duration of quarter notes equals to $\frac{60}{bpm}$s. Then we can get the duration of other beats due to the power of 2 relationship between the notes. A whole note is 4 (2 to the power of 2) times a quarter note, a half note is 2 times a quarter note, etc. The whole algorithm is given by Algorithm 1. In classical music the type of beats are common: The double whole-note, whole-note, dotted half-note (dotted means that attaching a dot means increasing half of the original duration. for example, if a half-note duration is 1s, dotted half-note is 1.5s), quarter

note, dotted quarter-note, eighth note, dotted eighth note, sixteenth note, dotted-sixteenth note, thirty-second note.

Algorithm 1 A Simple Algorithm To Further Processing The Frequency And Time Duration Series

Input: High-Residual files o_i for i $\in$ [1,M], where maximum i=M
Output: The frequency series f_i and the time duration series d_i of main melody line in o_i , where i$\in$[1,M].
for all $i \in$ [1,M] do
Translate file o_i form a High-Resolution format (DSD, DSF or FLAC)into WAV format $o_i^{'}$
Compute the spectrum of $o_i^{'}$, and get the total length t_i ;
Cut the spectrum into t_i/0.025 pieces and trace the main melody line l_i by separate the component frequency with the maximum amplitude in each piece;
Peak all frequency values as the frequency series.
Do Onset detection to the main melody line to get timestamp of every note with start moment in the main melody line;
Forward difference the timestamp series, get note duration series.
Do librosa.beat.tempo to get bpm bpm_i for each $o_i^{'}$
end for
Build standard pitch dictionary pitches = {g:$16.352 * 2^{\frac{g}{12}}$}, $\forall g = 0, 1, \cdots, 120$
Build standard pitch interval dictionary pintervals = $\{g : (16.352 * 2^{(\frac{g}{12}-\frac{1}{24})}, 16.352 * 2^{(\frac{g}{12}+\frac{1}{24})})\}, \forall g = 0, 1, \cdots, 120$
for $i \in$ [1,M] do
for all n in f_i do
if $n \in pintervals[g]$ then
if n = pitches[g] then
PASS
end if
if $n \neq pitches[g]$ then
n $\leftarrow$ pitches[g]
end if
end if
end for
Count the quarter note beat duration $t_i = \frac{60}{bpm_i}$
for all m in d_i do
Match every equivalent note beat by power of 2 relationship between the notes, quarter note duration as the basis.
end for
end for

Time series to supervise processing

After getting those frequency and note beat series, we treat them as time series. We first concatenate those frequency series into a whole series, the same as beats series. Then we use

those series to build labels.

It needs 4 steps: First copy the data series, and use this copy as labels. The pandas [66] provide a shift function and can help us shift all the observations of the raw data series down by a one-time step by inserting one new row at the top, set 0. The length of the series is n+1 if the raw series length is n, so we need to add a 0 to the tail of the label series, for keeping data series and label series to be the same length. Finally, we slice the entire sequence, each with a length of 20, and delete the extra fragments whose length is less than 20 at the end, and we reshape the series from [n,1] to [int(n/20), 20, 1]. We choose 80% of data and label for model training (in each epoch, we randomly sample 20% for validation, those sampled data is not be tained in the epoch), 20% left for generation.

This label processing method is also recorded in [67] can easily build time relation between label and data. The reason we choose this method is as following:

1. For the convenience of processing. Usually, researchers use the rolling method to handle time-series data. They select a fixed-length window, let the length is m, scanning the data from front to back, which has a length n, and copy the data covered by the window each step forward, and finally get the data with shape (n-m,m,1). And for deep learning, they set the last data in every window as label series with shape (n-m,1), as a many-to-one mode. But for our music time series, the length exceeds about 2.4 million, use the rolling method cost too much time and may cause a memory problem.

2. Another reason is that the many-to-one mode as just mentioned. Although music can be regarded as sequence data, which usually is described by a joint probability,

$$\begin{aligned} p\left(x_{1:T}\right) &= p\left(x_1\right) p\left(x_2 \mid x_1\right) p\left(x_3 \mid x_{1:2}\right) \cdots p\left(x_t \mid x_{1:(t-1)}\right) \\ &= \prod_{t=1}^{T} p\left(x_t \mid x_{1:(t-1)}\right) \end{aligned} \tag{3.1}$$

where T is the length of the sequence. Probability of note be chosen at time t, can be determined by all note in the given sequence $x_{1:t-1}$.

But we cannot apply this mode in music, which could build a wrong time relation. The length of a rolling window should be determined by motif (see Chapter 2.1.4), the smallest

”meaningful” unit of the music. So the many-to-many mode is better. What’s more, there are usually more than one motifs in a piece of music, so it’s arduous to find all motive sequences in a piece of music by programming.

In order to overcome this to a certain degree, we chose the above-mentioned labeling process. By shifting forward the copy of data to be the label, it produce a one-to-one time relative between data and label. Then cut into piece with each length is 20, producing many-to-many time relative mode for each data piece x_i and label piece y_i , $\forall i = 1, \cdots, int(n/20)$, n is the length of raw data sequences.

This is the entire process of obtaining the training set and making labels. We now begin to introduce the model part.

3.2 The model designation

In this section we focus on how the neural network model is designed. We design a deep learning model by combining the CNN and attention mechanism. We start with the structure with RNN attention model mentioned in Bahdanau’s work [31], who used the attention model to improve recurrent neural networks long-term dependence problem, and got a good result.

3.2.1 RNN-Attention in Bahdanau’s Work

We first trace back the RNN-Attention model based on the Bahdanau’s work. And there is a segment simple example code of self-attention by David Foster [68]. The model is built with Tensorflow [69]:

```
x = LSTM(units, return_sequences=True)(x) ①
e = Dense(1, activation='tanh')(x)
e = Reshape([-1])(e) ②
alpha = Activation('softmax')(e) ③
c = Permute([2, 1])(RepeatVector(units)(alpha)) ④
c = Multiply()([x, c]) ⑤
c = Lambda(lambda xin: K.sum(xin, axis=1), output_shape=(units,))(c) ⑥
```

$$notes\_out = Dense(notes, activation = 'softmax', name = 'pitch')(c)$$ ⑦

The explanation of above steps are as follows:

① The x denotes the last LSTM layers that are used as the recurrent part of the network. Set return_sequences to True to make each layer pass the full sequence of hidden states to the next layer, rather than just the final hidden state. The LSTM layers are to build the feature query tensor.

② The alignment function is a dense layer with only one output unit activated by a tanh function. We can use a reshape layer to squash the output to a single vector, of length equal to the length of the input sequence (seq_length), this reshape layer is optimal. From ②, we are aimed to build a key tensor for information aggregation by multiply .

③ The weights are activated by a softmax activation to the alignment values.

④ Use RepeatVector layer provided by Tensorflow to copy the core key tensor rnn_units times to form a matrix of shape[rnn_units, seq_length].Then reshape the repeated key tensor to [seq_length, rnn_units] though a permute layer.

⑤ In order to aggregate information, multiplying the repeated key tensor with the query tensor.

⑥ Finally, we use a Lambda layer to perform the summation along the seq_length axis, to give the context vector of length units.

⑦ Use a dense layer with notes units to output a sequence with length dense.

We now introduce our final model through step-by-step improvements.

3.2.2 Convolution with Bahdanau's attention

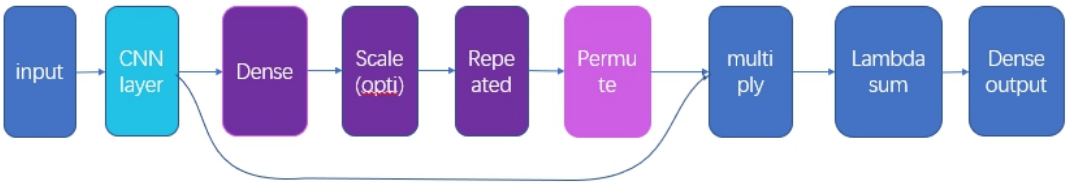


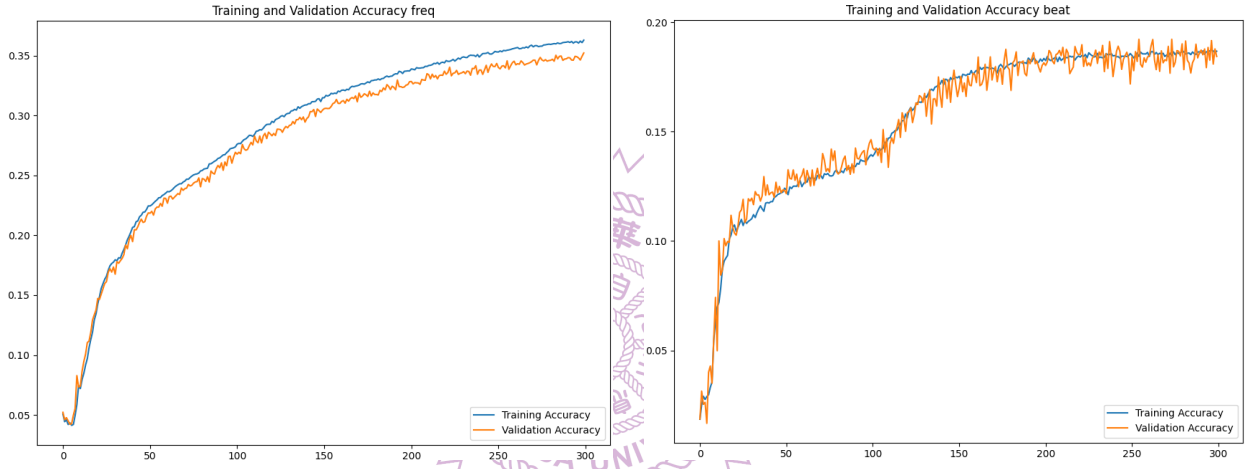
Figure 3.1: The structure of Convlution layer with Bahdanau's attention model

The ② - ⑥ are the steps to build the mechanism of Bahdanau's attention. We replace the

RNNs (①) with a convolutional layer to build a query tensor for the convolutional that can be processed parallized. Figure 3.1 shows the the structure.

In fact, a convolution layer also has the network capacity (see Chapter 2.3.4) as the RNN, LSTM layer does. Recall Eq.(2.6), if the area of convolution filters are overlaps, the last filter has some memory to the previous one. Another reason we use CNN for replacing RNN is inspired by the COCONet [25]. The training result is shown in 3.2, indicates that the model cannot learn the data well. For pitch, the model only gets 35% accuracy, and 18% accuracy for beat data. After 200 epoch training.

Our accuracy is defined by how often predictions equal labels. For our task, the accuracy for



training accuracy of pitch data training accuracy of beat data

Figure 3.2: The results of the convlution Bahdanau's attention..

pitch data in [0.6,0.85] is fine. Accuracy lower than 0.6, the model might not learn the pitch pattern, larger than 0.85 the risk of copy occurs, means the model output an almost the same sequence from the raw data.

3.2.3 CNN based time-distributed Bahdanau's attention

We then modified the step ② - ⑥ in Chapter 3.2.1, to get a better result. The new attention structure are based on the CNN, steps are:

x = Conv1D(filters=filterss, kernel_size=k,activation='relu',kernel_initializer='random_normal', bias_initializer='random_normal', padding='same')(input) ①

a_s=GlobalAveragePooling1D(data_format='channels_last')(x) ②

key = Permute([2, 1])(TimeDistributed(Dense(seq_len, activation='relu', kernel_initializer='RandomNormal', bias_initializer='RandomNormal',))(Reshape([filterss,1])(a_s))) ③

att = Lambda(lambda xin: keras.backend.sum(xin, axis=1), output_shape=(seq_len,))(Multiply()([x, Activation('softmax')(key)]))④

notes_out = Dense(notes, activation = 'softmax', name = 'pitch')(att) ⑤

The explanation steps are as follows:

① The CNN is used to extract features of the input data. We choose the 1D convolution layer for the data training. Set random_normal to kernel_initializer, and bias_initializer, for which the model can learn the dataset faster. We set a local activation with the relu function to keep the output has the same shape of input, we set 'same' to the padding parameter.

② We use a GlobalAveragePooling1D layer rather than a dense layer to produce a core key tensor. It calculates the average among the specified dimension as the channel. The output is a 1D sequence tensor with the length equals to the feature of the dimension. This layer can be also regarded as an encoder.

③ We use a TimeDistributed layer to replace the RepeatVector layer for combined the core keys. Then use the permute layer to let the tensor has the same tensor shape as the CNN tensor for aggregating information.

④ Then multiply the CNN layer and the produced key tensor in ③ to aggregate the information. the key tensor has been activated by softmax before multiply operation. We use a Lambda layer to perform the summation along the seq_length axis, to give the context vector of length units.

⑤ Use a dense layer with notes units to output a sequence with length dense.

Our improvement of the model in Chapter 3.2.2 has a certain effect on the data learning. For

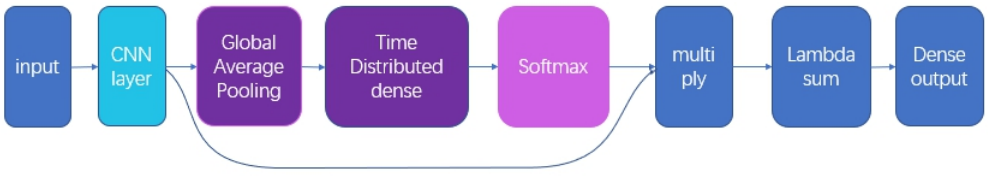
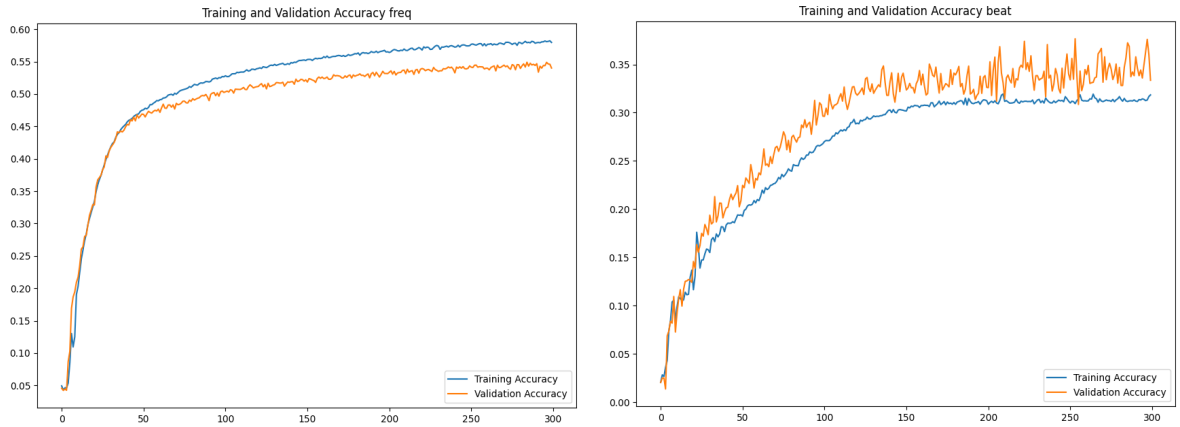


Figure 3.3: The structure of our CNN based TimeDistributed attention

pitch, model get 60% accuracy, and about 30% accuracy of beat data. After 200 epoch training. The structure is shown in Fig. 4.1. The result are shown in Fig.3.4. We explain here the reason



training accuracy of pitch data　　training accuracy of beat data

Figure 3.4: The results of our CNN based TimeDistributed attention.

why the model gets a better result. The attention mechanism can be regarded as a searching engine. The raw Bahdanau's attention, recalling ②-⑥ in Chapter 3.2.1, performance a precise matching, containing less information because of the dense layer(②), while our model contains more information since it has more data features for the Global average pooling layer.

3.2.4 A more complex model

We call the step ①-③ a CNN-attention block, then we plan to enhance the model learning ability by simply connecting two blocks directly (It needs a reshape for the summation layer output, which is a 2-D tensor with shape (n, seq_len), needs reshape to (n,seq_len,1). Because the input tensor of CNN-1D requires 3-D shape). The structure is shown in Figure ??. But a degradation occur.The accuracy rate of frequency is reduced to 35%, which is no better than just using one block. And for beat data, the accuracy is only 25%. After 300 epoch training. Result is shown in Figure 3.6.

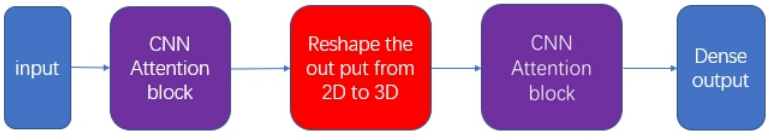
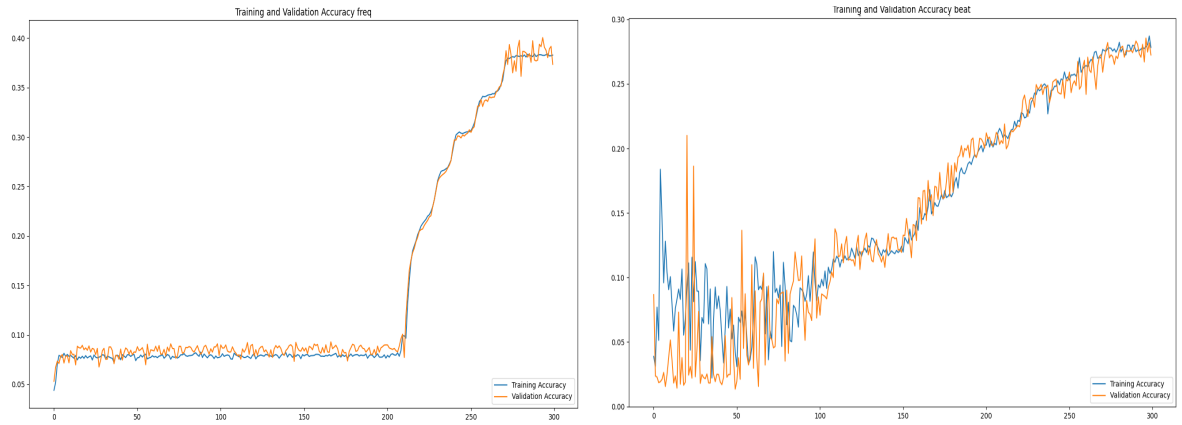


Figure 3.5: The structure of our 2-block CNN attention model, reshape conected



training accuracy of pitch data training accuracy of beat data

Figure 3.6: result of our 2 block CNN attention model..

We than replace the reshape layer between the 2 CNN-attention block with the TimeDistributed layer.See Figure 3.7. It performs better than the one only connected with a reshape layer especially on the beat data, the accuracy is improved to above 30%, but for pitch frequency data, the accuracy is reduced to 35% . Results are shown in Figure 3.8. It's hard to explain without

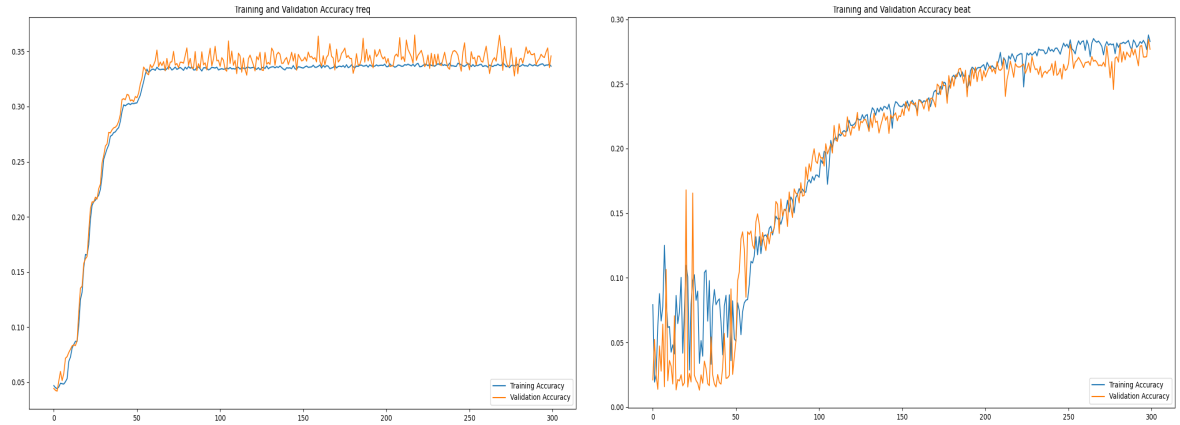
Figure 3.7: The structure of our 2-block CNN attention model, with TimeDistributed dense connected

obviously mathematic evidence, but we still try to give 3 possible reasons here to explain it:

1. Information is distorted between blocks. Notice that last operator is a summation operation which reduces the tensor dimension. Obviously, the operator drops some information in the query-key matching operations in the previous layers. This affects the second CNN-attention block performance.

2. What's more, information distortion causes an exposure bias problem [70]. Model prediction at time t depends on the previous predictions at 1:t-1. But the distribution of model $p_\theta(x_{1:(t-1)})$ is not equivalent to the real data distribution $p_{real}(x_{1:(t-1)})$. The error exists in the prediction of previous sequences $\hat{x}_{1:t-1}$ would propagate so that the generated sequence at or after time t will also deviate from the true distribution.

3. No previously attention model is mentioned like this connect two attention layers vertically.



training accuracy of pitch data　　training accuracy of beat data

Figure 3.8: Results of our 2 blocks CNN attention model with a TimeDistributed dense connection.

But in Vaswani's work [1]. They provide a multi-head model that uses several scaled-dot attention to process the data at the same time.

3.2.5 Further improvement

Usually, the solution to the degradation problem is to add some shortcuts to build a deep ResNet [33]. We are inspired by this and improve the model ability by approximating the scaled dot-product attention. This attention is also called self-attention/Intra-attention, which uses Query-Key-Value mode (as in Chapter 2.3.4), as shown in Fig. 3.9.

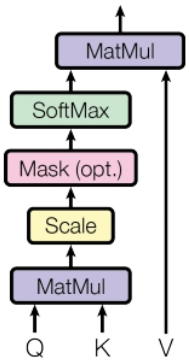


Figure 3.9: The Scaled Dot-Product Attention [1]

An explanation about the scaled dot-product attention

For example let X=$[x_1, \cdots, x_n] \in \mathbb{R}^{D_x \times N}$ be input, and the output be H=$[h_1, \cdots, h_N] \in \mathbb{R}^{D_v * N}$, the steps to compute the model are:

1. $\forall x_i \in X$, we first map it linearly to three different spaces, get the query vector $q_i \in \mathbb{R}^{D_k}$, key vector $k_i \in \mathbb{R}^{D_k}$, and the value $q_i \in \mathbb{R}^{D_v}$. The linear mapping can be simplified as:

$$\begin{aligned} Q &= W_q X \in \mathbb{R}^{D_k \times N}, \\ K &= W_k X \in \mathbb{R}^{D_k \times N}, \\ V &= W_u X \in \mathbb{R}^{D_v \times N}, \end{aligned} \tag{3.2}$$

$W_q \in \mathbb{R}^{D_k \times D_x}, W_k \in \mathbb{R}^{D_k \times D_x}, W_v \in \mathbb{R}^{D_v \times D_x}$, are the parameter matricies to be learnt, Q = $[q_1, \cdots, q_N]$, K = $[k_1, \cdots, k_n]$, V = $[v_1, \cdots, v_N]$, are matricies composed of a query vector, a key vector, and a value vector respectively.

2. For every query vector $q_n \in Q$, use Query-Key-Value attention, with output h_n,

$$\begin{aligned} h_n &= \text{att}((K,V), q_n) \\ &= \sum_{n=1}^{N} \frac{\exp(s(k_n, q))}{\sum_j \exp(s(k_j, q))} v_n \\ &= \sum_{j=1}^{N} \alpha_{nj} v_j \\ &= \sum_{j=1}^{N} \text{softmax}(s(k_j, q_n)) v_j \end{aligned} \tag{3.3}$$

where $n, j \; in [1, N]$ represents the position of the output and input vector sequence, α_{nj} represents the weight of the "concern" between the n-th output and the j-th input. Vaswani et al, use the scaled-dot product attention scoring function (see Eq. 2.19 in Chapter 2.3.4) as $s(k_n, q)$, can be written as :

$$H = V \, \text{softmax}\left(\frac{K^\top Q}{\sqrt{D_k}}\right) \tag{3.4}$$

softmax$(\cdot)$ is the activation function. The output, in fact, can be regarded as a probability, evaluating the relevance between the query vector $q_i \in \mathbb{R}^{D_k}$ and the key vector $k_i \in \mathbb{R}^{D_k}$. When

the input tensor has a high dimension, the output value of dot product attention (see Eq.(2.18)) may has a relatively large variance, which leads to the gradient of the Softmax function to be relatively small. To avoid this, they scaled the result of $K \times Q$ by divided $\sqrt{D_k}$. And besides, If K=V, Query-Key-Value is equivalent to the Bahdanau's attention [31].

Our data is only one dimension(a 1-D pitch sequence and a 1-D beat series), so we not need to scale our data by it's dimension.

First Q-K-V mode attention locally approximation

We provide the first improvement like this: To locally approximate the Q-K-V mode attention, to robust the connection between the 2 CNN-attention blocks. The query is the first CNN, the key is the TimeDistributed dense layer. Then, we multiply the TimeDistributed dense layer that connects the 2 CNN attention blocks and the raw input data(as value), the multiplied tensor then as the input of the second CNN. The structure of the model is shown in Figure 3.12. $\otimes$ denotes the multiply of two layers.

The result is meet our expectations after improvement. For frequency note, the predict accuray

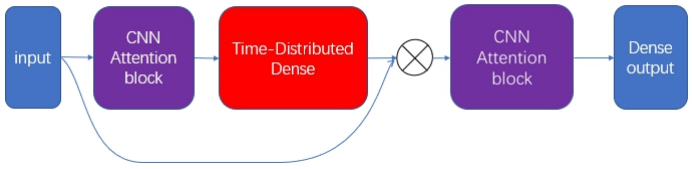
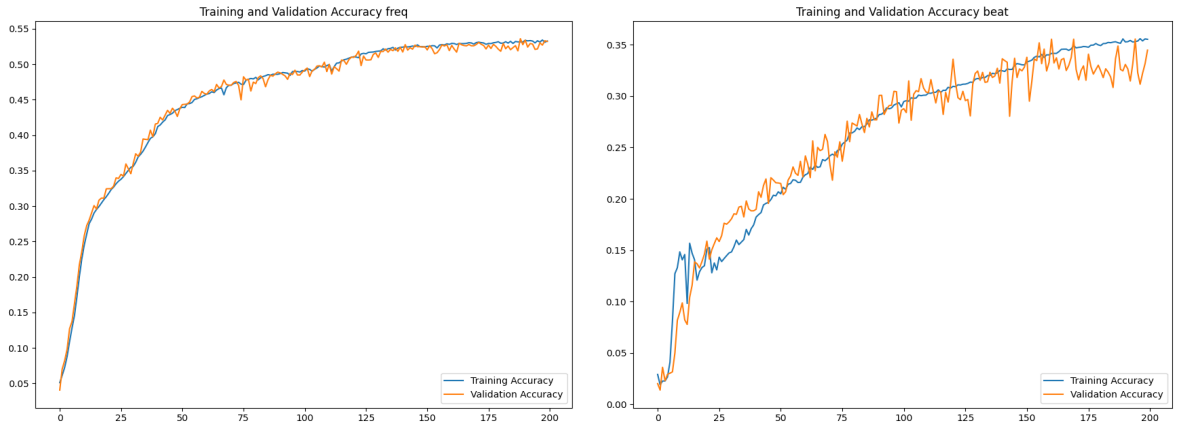


Figure 3.10: The structure of our attention with residual block model

is about 55%, accuracy of beats data rise to 35%. The improvement solve the degradation.

Second Q-K-V mode Attention Approximation

The second improvement is: we use the convolution layer in the second block as key, the TimeDistributed dense layer in the first block as the query tensor, and directly regards the input as a value tensor, to approximate self-attention. The structure of the model is shown in the Fig. 3.12. Network layers are denoted by ① : Convlution 1D; ② : Global average pooling 1D;③ : Reshape ; ④ : TimeDistributed Dense; ⑤ : Permute(optimal); ⑥ : softmax; ⑦ : lambda sum(recall Chapter 3.2.1,and Chapter 3.2.3). $\otimes$ denotes the multiply of two layers.



training accuracy of pitch data　　training accuracy of beat data

Figure 3.11: Results of our residual attention block model.

This improvement is not better than the previous one (as shown in Figures 3.12 and 3.8), for

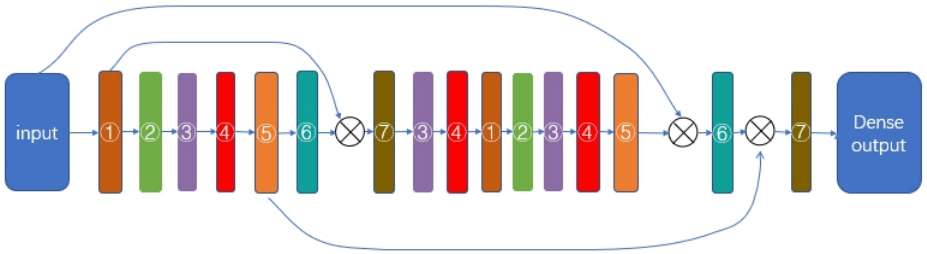


Figure 3.12: The structure of our locally Approximation self-attention model

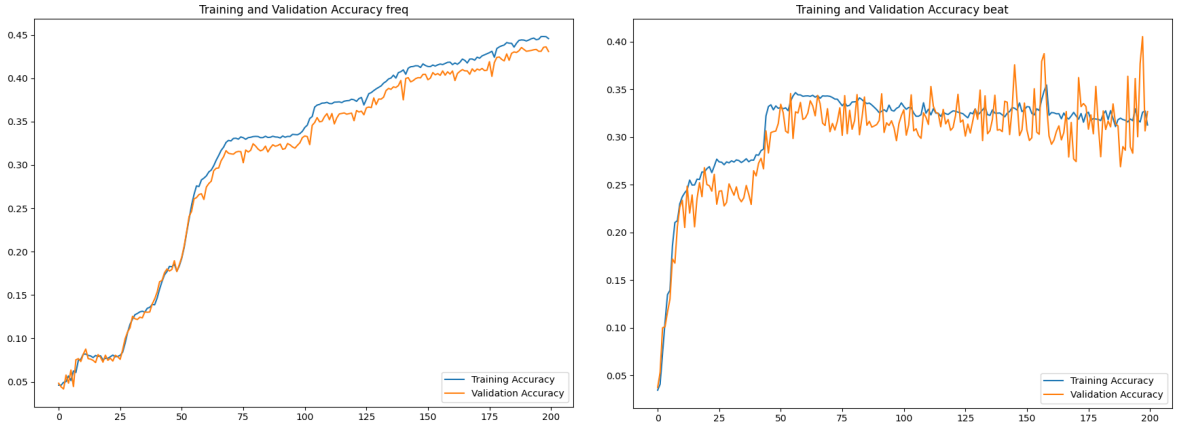
pitch data, the accuracy is only about 45%, the beat data is around 35%, results are shown in Fig.3.13.

3.2.6 The final model

We combined the two improvement models in Chapter 3.2.5, together, and get the best model, the final structure is shown in Figure 3.14.

For pitch data, the accuracy reaches 77%, and the beat data is about 41%, results are shown in Fig. 3.15. Beat data is difficult for model training. The beat is relatively solidified, and there is little change within a piece of music, but the beat pattern of different tracks may be completely different. That causes the training difficult.

The biggest advantage of our model is the efficiency. The number of parameters should be trained is so small that it becomes very fast, and costs less time and computing resource. This will be shown in next chapter.



training accuracy of pitch data training accuracy of beat data

Figure 3.13: Results of our local Approximation self-attention model .

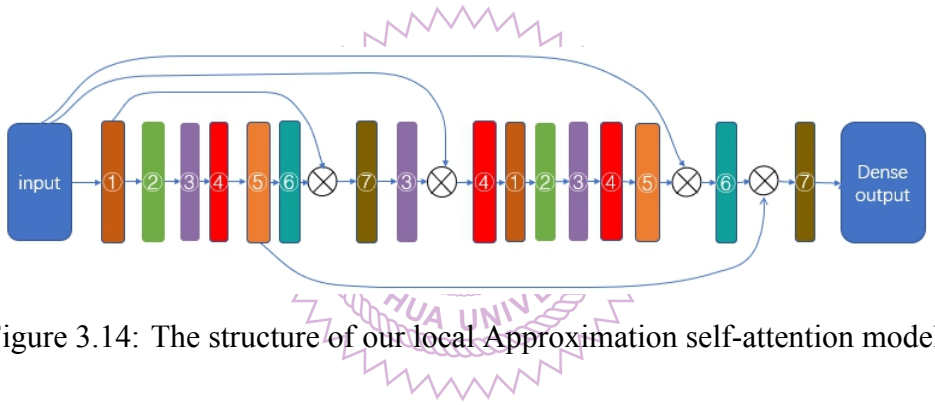
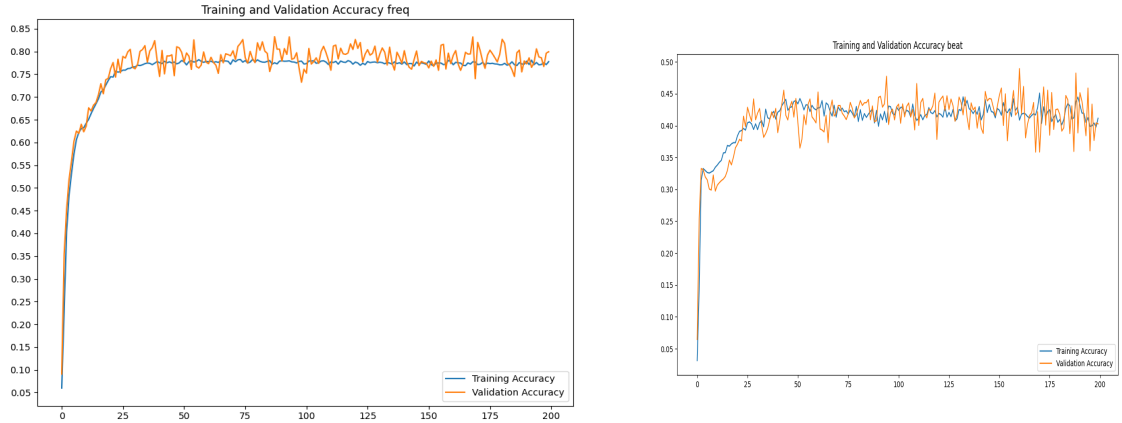


Figure 3.14: The structure of our local Approximation self-attention model



training accuracy of pitch data training accuracy of beat data

Figure 3.15: Plots of train accuracy of our the final model .

For the key, query, and value tensor, it's better to choose the tensor that contains more information or is closer to the input data(or the input itself). For the first block, CNN layer(①) and the TimeDistributed dense(④) layer should be key and query, the value tensor should be the input data or the 1st CNN layer; for the second block, CNN layer should be key, query and value could be: input data, CNN layer, TimeDistributed dense layer from the first block, or CNN layer, TimeDistributed dense layer from the 2nd block. There is a total of $2 * C_5^2 = 20$ different structures. We test 8 of them, and the structure is shown in the paper (model 1) get the best result (notice the model 6 get 0.06% higher than model 1 on the accuracy of beat, but it does not make any difference, and the accuracy of pitch series is obviously lower than model 1). We consider the accuracy of pitches is more important than the beats. All models have trained 200 epochs, with the same conditions (optimizer is Adam, with learning rate 0.008, the batch size is 3000).

We will introduce the evaluation of our proposed music generation models.

	1st K-Q-V mode	2nd K-Q-V mode	accuracy of pitch	accuracy of beat
model1(paper)	K: 1st ①, Q: 1st ④, V: input	K: 2nd ①, Q: 1st ④, V: input	77.00%	41.30%
model2	K: 1st ①, Q: 1st ④, V: 1st ①	K: 2nd ①, Q: 1st ④, V: 1st ①	75.80%	38.40%
model3	K: 1st ①, Q: 1st ④,V: input	K: 2nd ①, Q: 1st ④,V: 1st ①	74.33%	36.46%
model4	K: 1st ①, Q: 1st ④,V: 1st ①	K: 2nd ①, Q: 1st ④,V: input	76.46%	38.58%
model5	K: 1st ①, Q: 1st ④,V: 1st ①	K: 2nd ①, Q: 1st ④,V: 2nd ①	75.05%	40.20%
model6	K: 1st ①, Q: 1st ④,V: input	K: 2nd ①, Q: 1st ①,V: input	75.70%	41.36%
model7	K: 1st ①, Q: 1st ④,V: 1st ①	K: 2nd ①, Q: 1st ①,V: input	74.83%	41.22%
model8	K: 1st ①, Q: 1st ④,V: 1st ①	K: 2nd ①, Q: 2nd ④,V: 2nd ①	74.93%	39.61%
model9	K: 1st ①, Q: 1st ④,V: input	K: 2nd ①, Q: 2nd ④,V: input	76.50%	36.73%
model10	K: 1st ①, Q: 1st ④,V: input	K: 2nd ①, Q: input, V: 2nd ④	71.61%	36.41%

Table 3.1: The result of the models are been trained.

Chapter 4

Experiments and Evaluation

In this section we are focus on the evaluations of the model and the model's outputs. We do evaluations through three aspects: traning efficiency, objective evaluation by statistical observations, and subjective evaluations. To do comparison, we let our model and another CNN based binary directional LSTM (CNN-BiLSTM) model generate new sample pieces, then use absolute metrics and relative metrics to observe the raw data-set and output set of those model. The CNN-BiLSTM model also performance well, the accuracy of pitch also reach 75%, about 43% of beat data.

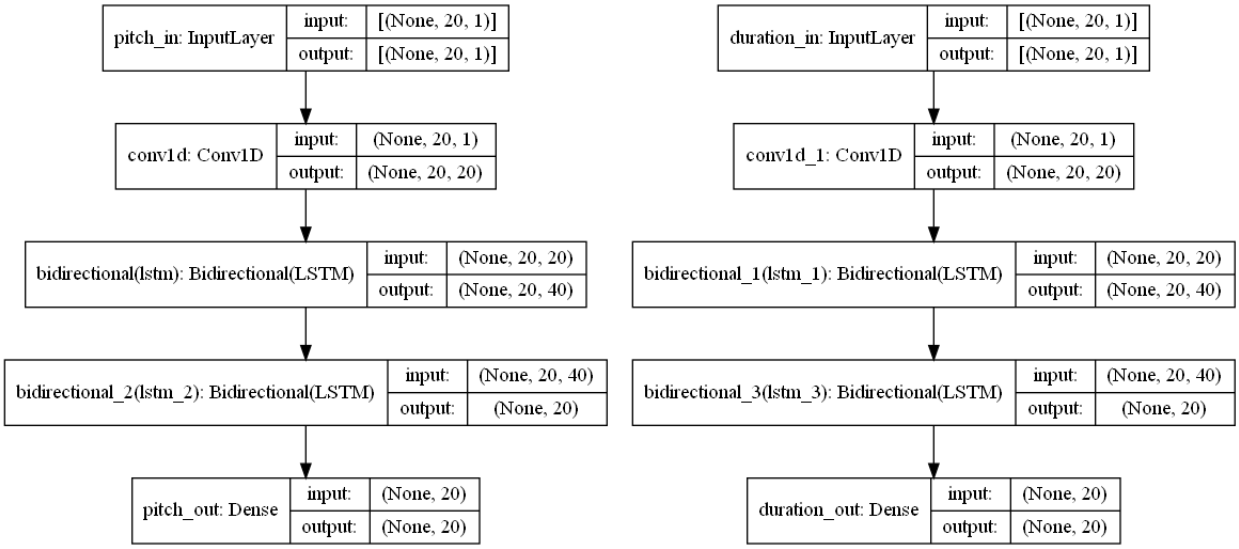
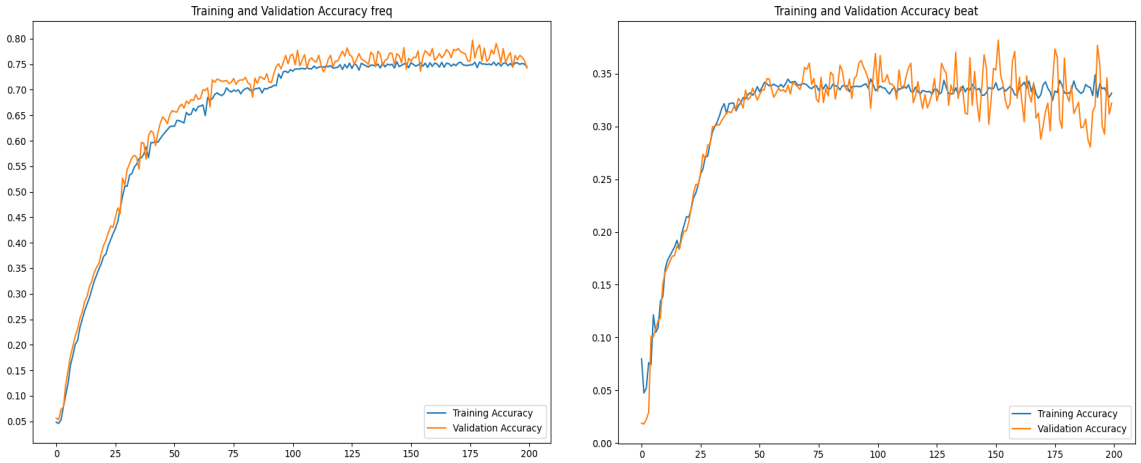


Figure 4.1: The structure of the CNN Bi-directional LSTM model



(a) training accuracy of pitch data (b) training accuracy of beat data

Figure 4.2: Plots of training accuracy of the CNN Bi-directional LSTM model

4.1 Training effiency evaluations

We can use the summary to check the parameter.Our model has only 8640 hyperparameters to be trained. However, the CNN-BiLSTM has 54560 hyperparametersto be trained. This is due to the advantage characters that shared weight and locally connection of the CNN. The number of hyperparameters is reduced.

4.2 A review of evaluations

The progress of AI composing itself far exceeds the evaluation work for AI-generated samples. Some music AIs are now able to generate works similar to human composers, but the evaluation of music samples has many obstacles. There are objective and subjective processes for evaluating the music generated by AI. Since the final say of the creative output is the person (listener or audience), subjective evaluation is usually the first choice in the generative model. However, looking at the previous subjective evaluation tests, there are so many factors that seriously affect the results [2].

Subjective evaluation: These evaluations either follow the concept of a musical Turing test, whose purpose is to make the subjects to distinguish which musical pieces are composed by people or by computers. If a higher proportion of subjects cannot distinguish（probability larger

than 80%）, the performance of the machine composition will be recognized as achieving the same ability as people. The summary of the experiments issues are as follows:

1.Confusion of purpose: One of the fundamental issues, however, is that many studies confront two questions on whether a music piece is aesthetically pleasing and whether it is composed by a human. [2].

2. Design of listening experiment: It comes the design of a listening experiment, which involves many aspects, and the environment control and the wording design of the questions should not interfere with the results of experiments. Besides, these listening assessments use different standards. Lacking standard comparison, it is unable to provide any absolute quality measurement which the proprietary questionnaires and listening examples make the results difficult to compare. Those results can't be regarded as representing a scientific basis.

3. Subjects: Most studies ignore the professional level of the subjects, and often overestimate the subjects' ability [2, 71], which affects the experimental results.

4. Statistical issue: Most research's sample sizes are so small that raise questions about statistical significance of the study [2, 72].

Objective Evaluation: The objective evaluation methods used in the latest research are classified into the following categories: 1. Probability measurements without knowledge of the music field; 2. Task/model-specific metrics; 3. Use measurement knowledge in the general music field. An example of the probability measurement is proposed by Huang et al [25], which is a frame-wise assessment method computing the negative log-likelihood between the model output and the ground truth across frames. The biggest problem is this kind of evaluation does not necessarily conform to human aesthetics since the lacking of the relative of the general music field, even the result in line with statistical performance. Task/model-specific metrics method such as the one proposed by Bretan et al [73], predicting a music unit from a pool of units in a generative system by evaluating the rank of the target unit. Those evaluation methods use custom-designed indicators instead of standard indicators, which may cause a danger of non-comprehensive assessment, which means that methods only work on a part or only several aspects of output, which would influence the efficacy of the assessment. It

should be pointed out that objective evaluation strategies should only be regarded as formative evaluations, which contribute to subjective evaluations, which are regarded as summative evaluations [2]. We combine objective evaluation and subjective evaluation for the generation samples of our model. Mainly use the method proposed by Yang and Lerch [2]. This is an example of use measurement knowledge in the general music field.

4.3 Objective music measurements With music features

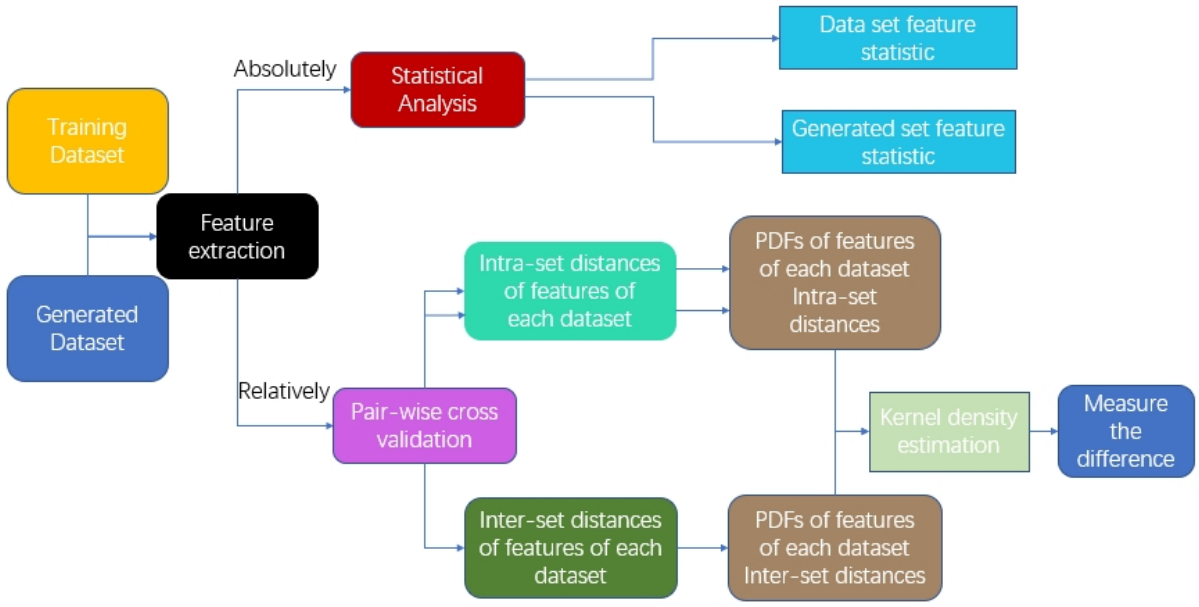


Figure 4.3: Evaluate flow by Yang and Lerch,2018, replot [2]

The two targets of the proposed evaluation strategies are to provide (i) absolute metrics to give insights into properties and characteristics of a generated or collected set of data, and (ii) relative metrics in order to compare two sets of data, e.g., training and generated.

Yang and Lerch extracted a set of custom-designed functions, which are rooted in the knowledge of the music field but are easy to understand and explain. We choose the core features of them, including pitch-based and rhythm-based features. Absolute measurement values can be calculated (Fig.4.3 top), and relative measurement values can be deployed after extracting these features. The absolute measurement can provide useful description of dataset properties. The relative measurement (Fig.4.3, bottom), allows to compare two distributions in various dimen-

sions. This measurement is computed by first applying pairwise exhaustive cross-validation to compute the distance of each sample to either the same dataset (intra-dataset) or to the other dataset (inter-dataset). The results are distance histograms per feature. Next, the probability distribution function (PDF) of each feature histogram is estimated by kernel density estimation.

Finally, Yang and Lerch compute two metrics for the objective evaluation of generative systems from the training dataset's intra-set distance PDF (target distribution) and the inter-set distance PDF between the training and generated datasets [2]: (i) the area of overlap and (ii) the Kullback-Leibler divergence (KLD). The steps are introduced in detail in the following sections. We also provide Jensen–Shannon divergence (JSD), can replace the KLD.

We randomly sampled 2000 pieces from the dataset as input without duplicate. Then asking the two model (our model,and the CNN-BiLSTM model) generate a new sample. Then, we compare the each generated set with the sampled dataset, and compare the generated set with each other.

4.3.1 Feature extraction

The features listed below are computed for both, the entire sequence, and for each measure in order to get some structural information. Those features can be divided into two independent parts: Pitch-based and Note-based, each of them do not contains other side information, Pitch-based features do not contains information from Note-based features, and vice versa.

Pitch-based features

Pitch count (PC): number of different pitches within a sample. The output is a series of scalar for each sample.

Pitch class histogram (PCH): it is an octave-independent representation of the usage of different octave-independent pitches.

Pitch class transition matrix (PCTM): The transition of pitch is a two-dimensional pitch class transition matrix is a histogram-like representation computed by counting the pitch transitions

for each (ordered) pair of octave-independent pitches.The resulting feature dimension is 12 × 12. These features contain useful information for evaluation tasks: it can be used for detecting melody activities, the 'tendencies and characteristics' in melodies. Furthermore, it shows the distinctive music styles, and can be used to classify two different music styles.

Pitch range(PR): The pitch range is calculated by subtraction of the highest and lowest used pitch in semitones. The output is a scalar for each sample.

Note-based features

Note count (NC): The number of used notes.

Note length histogram (NLH): To extract the note length histogram, we first define a set of allowable beat length classes and those note classes which mean and standard deviation of each feature of the data is computed. The double whole-note, whole-note, dotted half-note, quarter note, dotted quarter-note, eighth note, dotted eighth note, sixteenth note, dotted-sixteenth note, thirty-second note. This feature, which is similar to the PCH, represent the usage frequency of different note length.

Note length transition matrix (NLTM): This feature is similar to PCTM. It counts the pitch transitions for each (ordered) pair of different note lengthes.

features	dataset sample		our attention model output		The CNN-BiLSTM output	
	Mean	STD	Mean	STD	Mean	STD
PC	9.60600	3.066	14.147	2.51185	13.34	2.51185
PR	38.14	14.9353	52.168	25.7997	49.923	25.82822
PCH	-	-	-	-	-	-
PCTM	-	-	-	-	-	-
NC	5.33	1.2317	6.102	1.33251	5.614	1.10227
NLH	-	-	-	-	-	-
NLTM	-	-	-	-	-	-

Table 4.1: Result of absolute measurement .

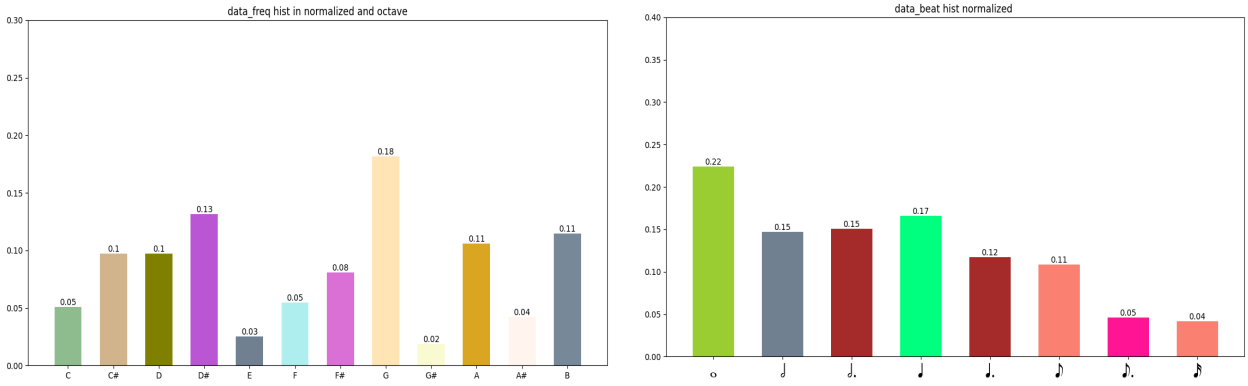
4.3.2 Absolute measurement

Table 4.1 shows the result of the absolute measurement of our dataset samples. Class histogram and class transition matrix for both pitch and note are not provided value since those features are normalized so that the summation of all components is 1. The class transition matrix and Class histogram are shown in the graph later. Absolutely measurement can roughly observe the features' action. PC and PR could be counted. Dataset test samples are sampled from the raw dataset series. Observing the result table, the generation of both models has a more diverse usage of pitch and note material since both feature means are larger than the dataset. Furthermore, a large difference in the mean or variance (STD) of the features of different sample sets can show that the sets are different in these indexes [2]. The difference of mean standard deviation can indicate that those models can generate materials of pitch and beat that do not exist in the sampled data. This also proves that the model does not give a sample copied from the dataset and it is generated by itself.

Now observing the histograms as in Fig.4.4 and transition matrix plots as in Fig.4.5. Those plots also show different usage trends of those models with the different pitches and beats. The frequency class transition matrix of our model shows a more diverse action than the CNN-BiLSTM model's. This also proves that the model does not give a sample copied from the dataset and it is generated by itself. But because the outputs are learned by the models from the same dataset, so the plots show similarity to some degree. For example, plots of beat transition matrix show the same area distribution of class transitions, although inside the area it has the difference.

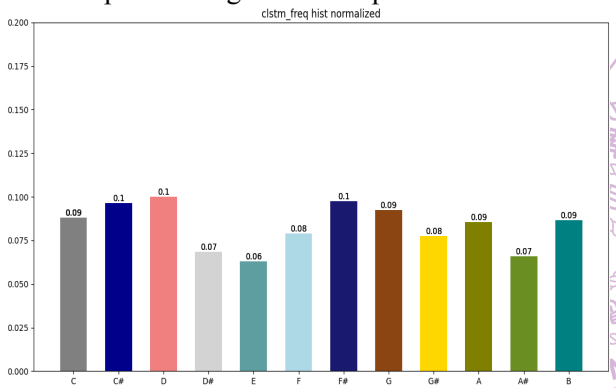
4.3.3 Relative Measurement

Relative measurement is a method of comparing different data sets. We first compare the sampled data set with the output sets of the two models and then compare the samples generated by the two models with each other. In relative measurement, the mean and STD of intra-set and inter-set of every features are computed, then observe the hidden probability dense function of

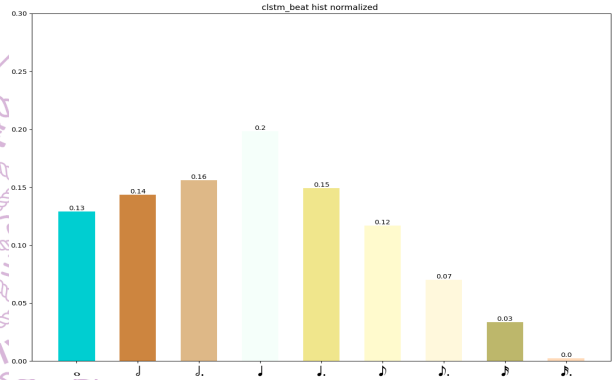


pitch histogram of sampled dataset

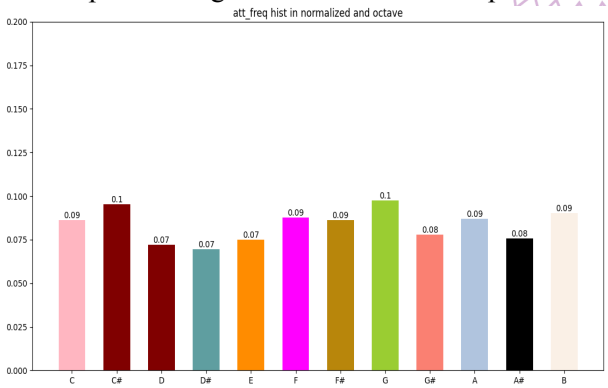
note histogram of sampled dataset



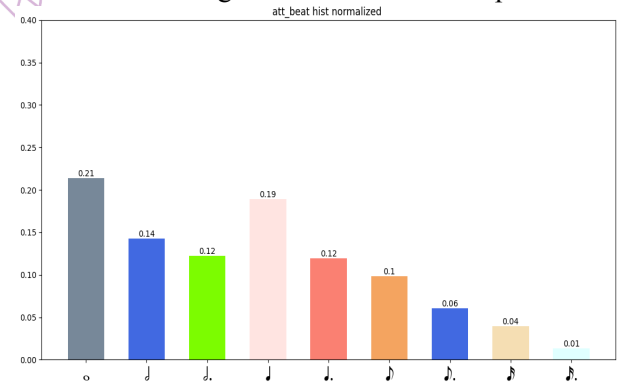
pitch histogram of C-BiLSTM output



note histogram of C-BiLSTM output

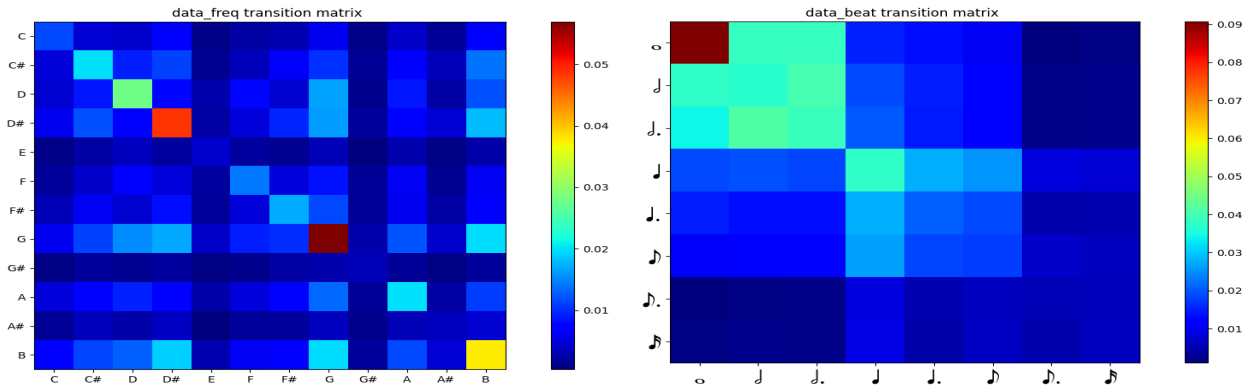


pitch histogram of our model's output

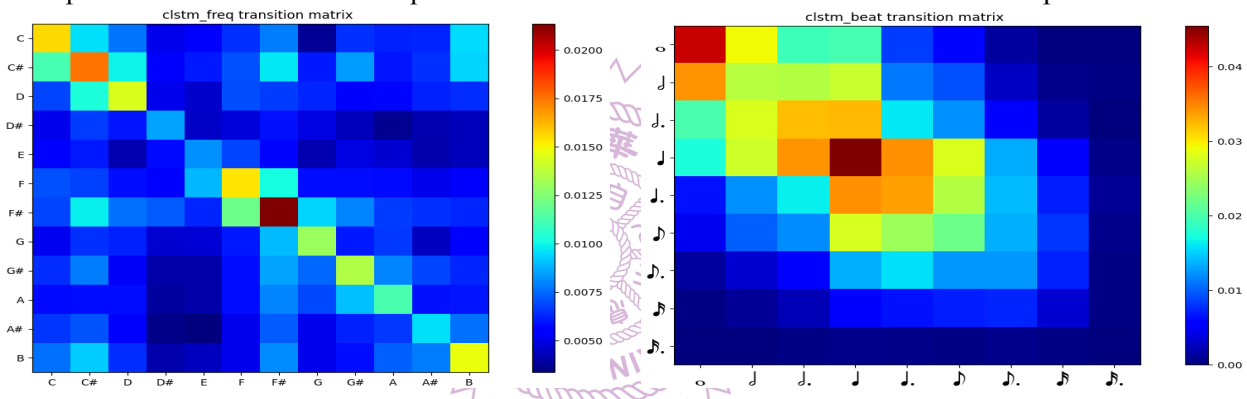


note histogram of our model's output

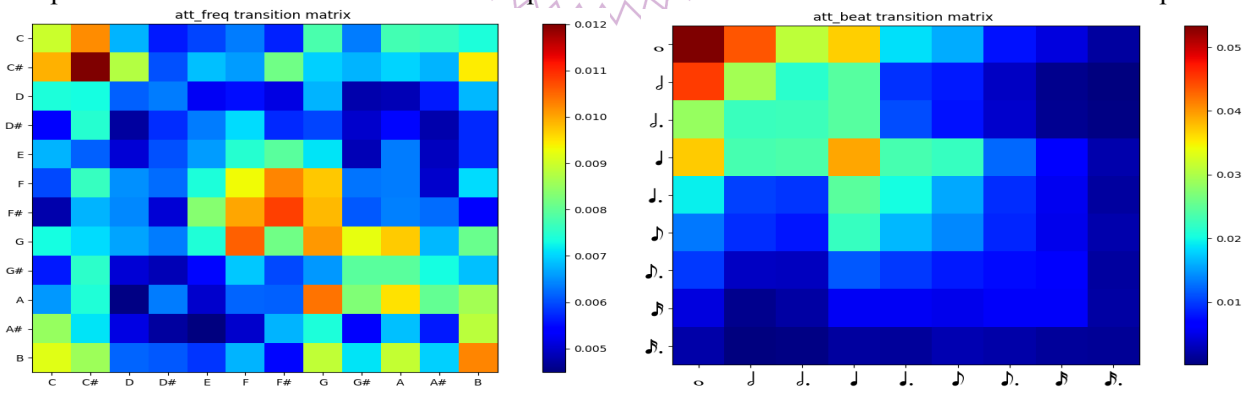
Figure 4.4: Result of histograms..



pitch transition matrix of sampled dataset　　note transition matrix of sampled dataset



pitch transition matrix of C-BiLSTM output　　note transition matrix of C-BiLSTM output



pitch transition matrix of our model's output　　note transition matrix of our model's output

Figure 4.5: Result of transition matrix..

those dataset, then provide KLD and OA of each feature to observe and compare the sampled dataset and model outputs.

features	dataset sample				The CNN-BiLSTM's output				Inter-set	
	Intra-set		Absolutemeasure		Intra-set		Absolutemeasure			
	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
PC	3.4061	2.68324	9.606	3.066	3.13372	2.42086	13.34	2.80007	1.17604	1.49765
PR	16.9024	12.66626	38.14	14.9353	27.05706	24.53792	49.923	25.82822	1.39321	2.30611
PCH	0.5545	0.2057	-	-	0.38916	0.1009	-	-	1.25558	1.55495
PCTM	0.4433	0.13608	-	-	0.35612	0.04921	-	-	4.60956	8.14696
NC	1.3463	1.10527	5.33	1.2317	1.20093	0.99387	5.614	1.10227	1.99563	3.276
NLH	0.5067	0.20728	-	-	0.46613	0.17407	-	-	1.4395	1.9553
NLTM	0.4173	0.11961	-	-	0.3833	0.07857	-	-	2.70568	3.60253

Table 4.2: Result of relative measurement of the dataset and CNN-biLSTM model's output.

features	dataset sample				Our model's output				Inter-set	
	Intra-set		Absolutemeasure		Intra-set		Absolutemeasure			
	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
PC	3.4061	2.68324	9.606	3.066	2.80145	2.18418	14.147	2.51185	1.79185	2.70349
PR	16.9024	12.66626	38.14	14.9353	27.72683	23.71646	52.168	25.7997	2.35843	3.63791
PCH	0.5545	0.2057	-	-	0.35531	0.08411	-	-	3.08392	6.90838
PCTM	0.4433	0.13608	-	-	0.33970	0.03486	-	-	4.9991	10.24791
NC	1.3463	1.10527	5.33	1.2317	1.48078	1.16554	6.102	1.33251	2.25448	3.0064
NLH	0.5067	0.20728	-	-	0.41423	0.15008	-	-	0.54712	0.35673
NLTM	0.4173	0.11961	-	-	0.36631	0.07923	-	-	4.68016	8.75521

Table 4.3: Result of relative measurement of the dataset and our model's output.

We first compare the sampled dataset with the outputs of the CNN-BiLSTM and our model, results are shown in table 4.2. And the comparison results of the sampled dataset and the output of our model are shown in Table 4.3. Comparing the result of intra-set and inter-set, mean values of the inter-set distances are larger than the mean values of both intra-set distances [2], especially for those feature's absolutely measurement means and deviations can be provided. This indecate that those features of datasets are different. For our work, this judgment method may be invalid for the histogram and class conversion matrix, the result does not meet the above judgment. But the graphs as in tables 4.4,4.5 show the difference between the sampled dataset and the outputset. We can also observe a similar result from table 4.2. Our model's performance shows a more diverse pitch and note usage.

For histogram and class transition matrices, we provide paired violin plot of the PDF of the

features' intra-set distances.A significantly higher skewness (the half plot of the dataset features PDF) indicates a less diversified intra-set behavior [2], as in Fig.4.6a and 4.6b. We can also use the plot to compare the outputs. On those four features, our model performs slightly better than the CNN-BiLSTM model as shown in Fig.4.6c.

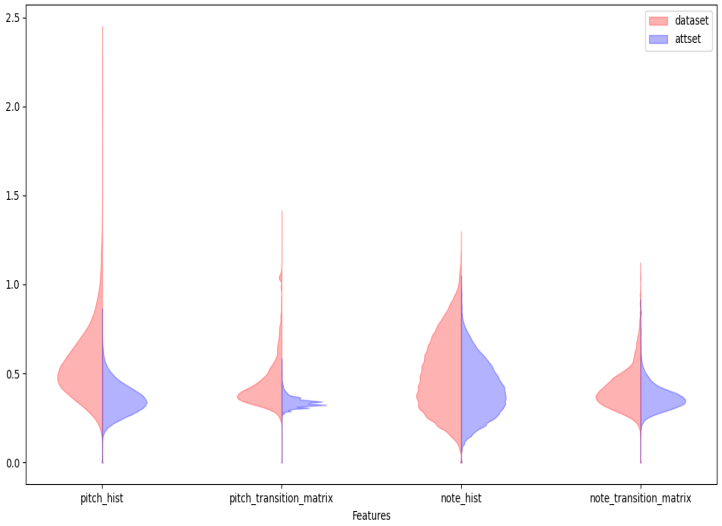
Then we evaluate the generation performance of the two models' output by distributions similarity. Similarity measures between distributions are also provided for evaluating the statistical measures representing intraset distances or inter-set distances. Yang proposed the method computing the KLD and OA, for KLD is unbounded and asymmetric, so they compute the OA, abounded measure in the range $\in [0,1]$.

As Fig.4.7a and Fig.4.7b show OA reduces and KLD increases from dataset to the each of outputs set for most features: PC, PH, PCTM, and NCTM. Those features approximately fit the result of features' intra-set distance, which has a slightly lower mean and STD than the dataset, which might indicate the loss of diversity. PR shows an opposite result: both OA and KLD increasing from dataset to the each of outputs. This reveals the limitations of KLD as visualized in Fig.4.7, the KLD is calculated element-wise, PDFs with identical shapes (as indicated by similar Kurtosis and Skewness) that yields an insignificant difference in KLD. Shown in Fig.4.8, the PDF curve proof the inference above. They are similar to each other. OA can also face with troubles and be misleading when PDFs vary in their kurtosis but with a similar mean [2].

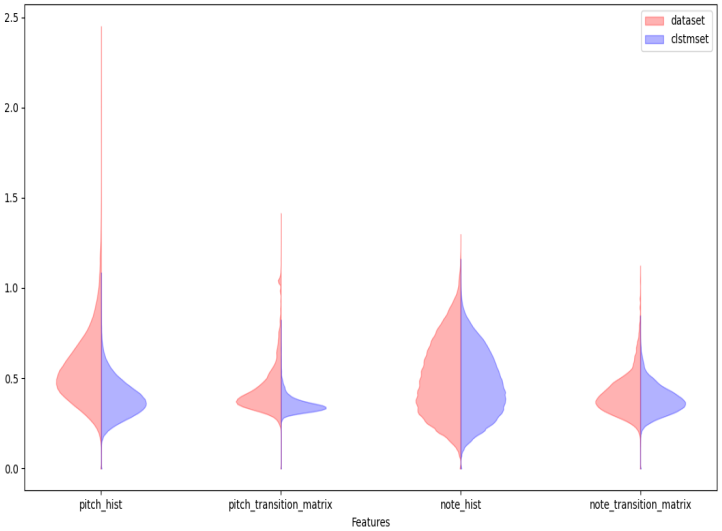
For NH, from the dataset to our model, OA increasing but reducing from dataset to CNN-BiLSTM . Both of the OA and KLD values for PR are quite close. This may reveal the PR is meaningless in our case. Besides, its STD is relatively high, which might hint at lower reliability of the mean value [2].

Another interesting observation is the note count (NC), which shows a result that both OA and KLD reduce from dataset to outputs. Both of the two models can output samples that contain the dotted-sixteenth note, which does not appear in the dataset (as in Fig.4.4 and Fig.4.5), and probably KLD and OA are facing the drawbacks of themselves on this feature.

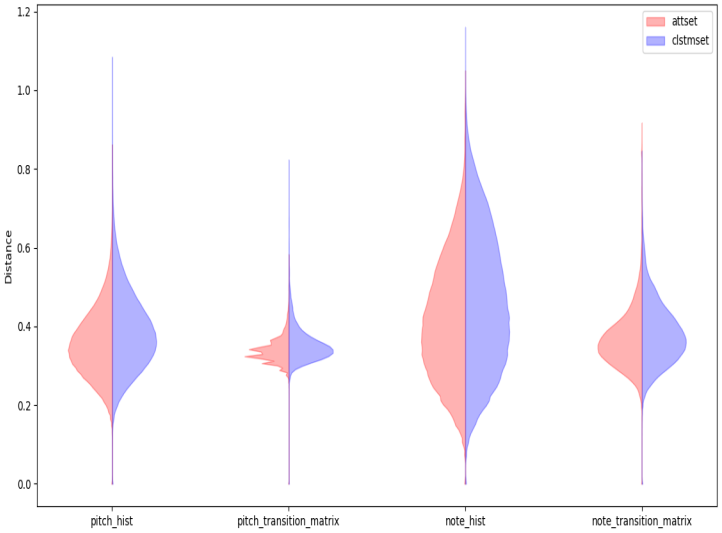
We plot the intra-sets and inter-set PDF of note count, as in Fig.4.8, for this feature, PDF of intra-sets and inter-set are too similar, which support our view to some degree. We can also



(a) Paired plot of sampled dataset (pink) and our model's (attset, blue) output

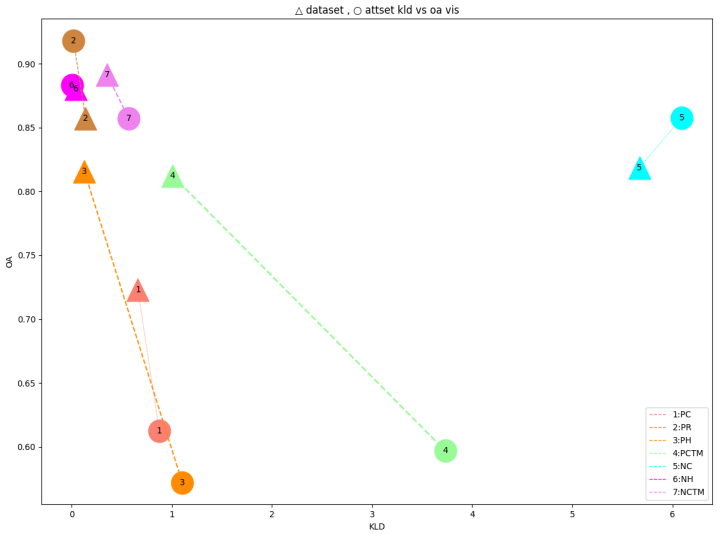


(b) Paired plot of sampled dataset (pink) and the CNN-BiLSTM's (cLSTMset, blue) output

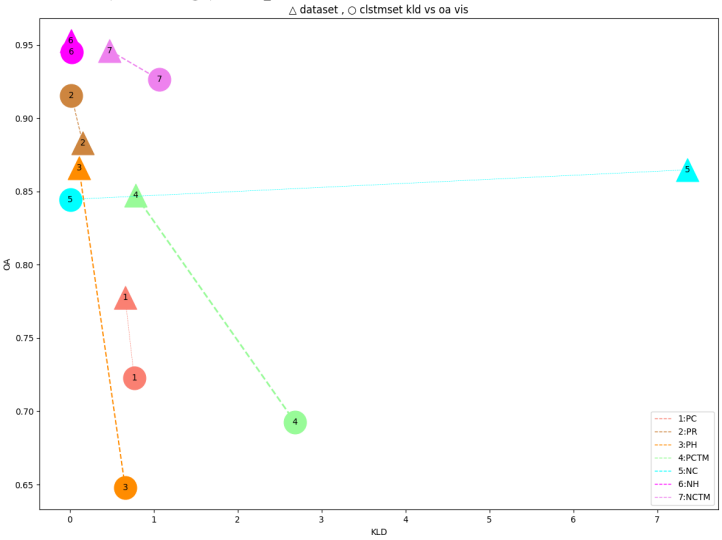


(c) Paired plot of our model (attset, pink) and the CNN-BiLSTM's (cLSTMset, blue) output

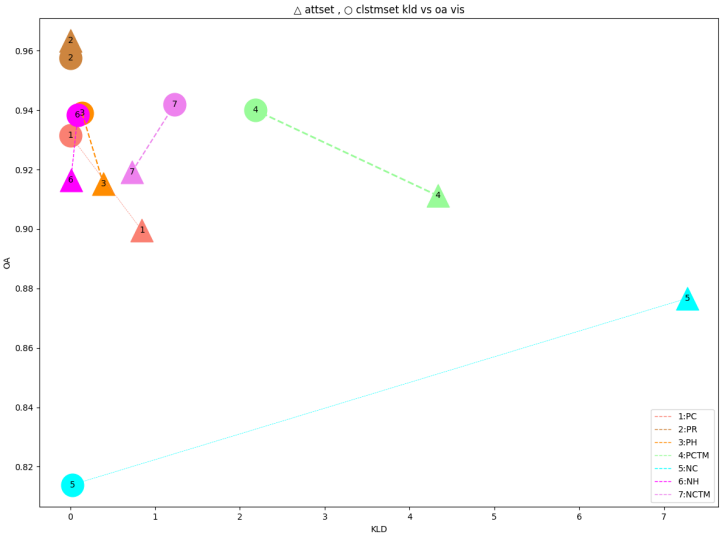
Figure 4.6: Result of paired violin plot of the PDF.



(a) KLD & OA plot of sampled dataset (△) and our model's (attset,◯) output



(b) KLD & OA plot of sampled dataset (△) and CNN-BiLSTM's (clstmset,◯) output



(c) KLD & OA plot of our model (△) and the CNN-BiLSTM's (attset,◯) output

Figure 4.7: result of distributions similarity,.

compare the outputs of these models by calculating OA and KLD. However, according to Yang and Lerch, having a larger OA and a smaller KLD on almost all features indicate that feature matching can provide the expected improvement [2]. The CNN-BiLSTM model performs better than our model on this point as shown in Fig.4.7c.

For the KLD is unbounded and asymmetric, we provide JSD to give a more evaluation

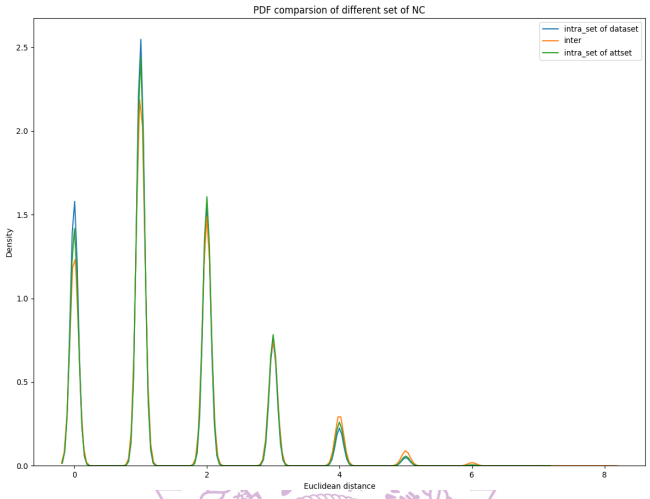


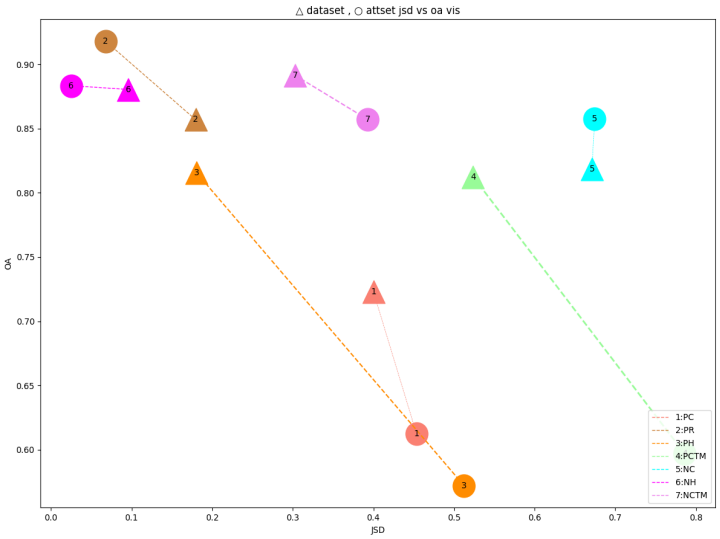
Figure 4.8: PDF plot of the NC feature of dataset and the CNN-BiLSTM's(clstmset)

reference, as in Fig.4.9. Because JSD also bounded in [0,1], it has better interpretation and comparison capabilities than KLD.

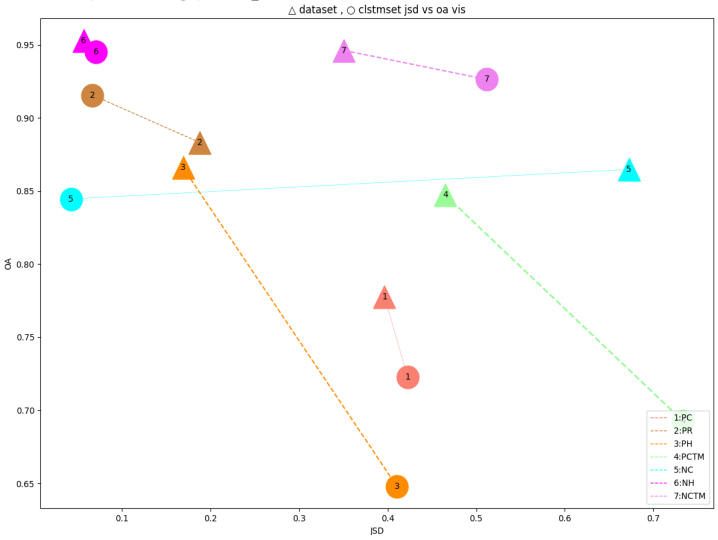
But JSD may also face with the same problem when measuring the distribution similarity on some features as KLD does, we visualize the OA and JSD in the same way as shown in Figures 4.9a, 4.9b and 4.9c.

4.4 Subjective Evaluation

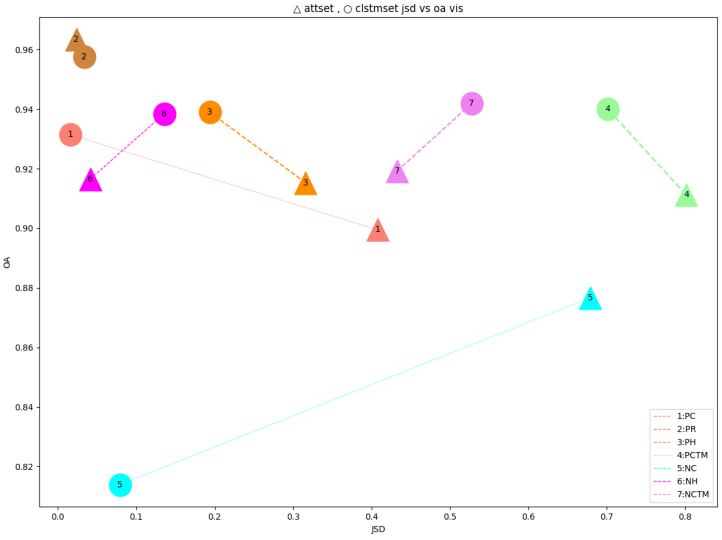
In objective evaluation, two models have their advantages and drawbacks. However, corresponding the statistical performance cannot say necessarily in line with human aesthetics. We design an online questionnaire for subjective evaluation, as shown in .A. Referring to the views proposed by Yang and Lerch on those subjective evaluations [2], we design a questionnaire based on music theory, strict but could be useful in general.



(a) JSD & OA plot of sampled dataset (△) and our model (attset, ◯) output



(b) JSD & OA plot of sampled dataset (△) and CNN-BiLSTM (cLSTM,◯) output



(c) JSD & OA plot of our model (attset,v△) and CNN-BiLSTM (cLSTM, v◯) output

Figure 4.9: Result of distribution similarity,.

4.4.1 Designation of the subjective evaluation questionnaire

We set constraints to avoid some situations that have occurred and may affect the results. First, we set a limitation to control who will answer our questions. Most people should have a music background since the questions involve professional music theory questions to evaluate the quality of the output samples. Then we restrict the environment (the surrounding objective environment, the internal subjective environment) when completing the questionnaire. We require subjects to be completed in a quiet and undisturbed environment. And they will not be affected by emotion when filling out the questionnaire. We consider that environmental and emotional interference may cause inaccurate questionnaire results.

For the question set, we first ask the personal interests and music identity (The favorite music style and musicians. These questions will not be included in the analysis and discussion, just for warm-up). Musical identity can be divided into three categories: 1. Professionally trained (called type 1 below); 2. Amateurs with musical knowledge (called type 2 below); 3. none of the above (called type 3 below). Only type 1 and type 2 joint is the best situation. A total of 31 people answered the questionnaire. Only 5 people belong to type 3. They have not been trained, so we can't count on them too much. We set two-part for the two models. One is for our our model, and the other one is for the CNN-BiLSTM model. We put samples from our model in Part 2. We give 2 samples in each part. Those samples are the best two from each model. For each sample, two questions are asked after hearing. We could not directly ask whether the sample is good since different people have different judgments of good music. Two aspects should be considered: 1. whether the sample follows the music theories; 2. whether the model influence the listeners' mood and thoughts. The first one asks the sample is whether structurally organized and stable changing, based on music theory (Q1 in each part). The second question asks whether the sample influences the inner mood and the thoughts of the participants (Q2 in each part).

After asking the above two questions for the two samples, in each part, we directly asked how are they feeling when hearing these samples of the two models (Q3 in each part), and conducted a Turing test to ask whether these two samples were not caused by human composition

(Q4 in each part). Each question is a grading question, the score range is $Z \in [1, 6]$, Z represents the integer set. 1 means the most inconsistent with the problem description, 6 means the most consistent with the problem description. We measure it according to the average value. The higher the average score, the better, except for the Q4 . It' s the lower the score, the better.

Finally, we still need the comparison of those model directly, and we ask which of the two model will they stand with after comparing those samples. That's the last question.

4.4.2 The Result Discussion

The results are shown in Table 4.4. We receive 31 answers, only 5 of type 3, other 26 all belong to type 2. Those codes are corresponding with the question codes in Appendix .A.

In fact, none of the participants chose the type 1. This would reduce the reliability of our

	S1Q1	S1Q2	S2Q1	S2Q2	Q3	Q4
result of all answer, Part1, C1.						
Mean	2.80645	3.38709	3.74193	3.32258	3.29032	4.35483
STD	1.077631	1.20214	0.96497	0.90873	1.16027	0.98483
result of all answer, Part2, C2.						
Mean	3.64516129	3.580645161	3.709677419	3.290322581	3.774193548	3.709677419
STD	0.950381927	0.992444576	1.160274346	0.937853876	0.762000762	0.824360293
result of type 2, Part1, C1.						
Mean	2.7	3.4	3.6	3.1	3.25	4.35
STD	1.086985953	1.19807538	0.941357449	0.919029589	1.176696811	1.017538508
result of type 2, Part2, C2.						
Mean	3.6	3.45	3.55	3.1	3.65	3.75
STD	0.986836437	1.017538508	1.164209867	0.928190962	0.7493587	0.745241314
result of type 3, Part1, C1.						
Mean	3.4	3.6	4.4	3.6	3.6	4.4
STD	0.894427191	1.341640786	0.894427191	0.894427191	1.140175425	0.894427191
result of type 3, Part2, C2.						
Mean	4	3.2	4	3.2	3.6	4
STD	0.707106781	0.836660027	1.224744871	1.095445115	0.894427191	1.224744871

Table 4.4: The result of Part1.

results. That need to be clarified first.

The Mean value of each question is the measurement of the scoring result. The standard deviation is used for observation assistance, which may show the reliability of the answers.

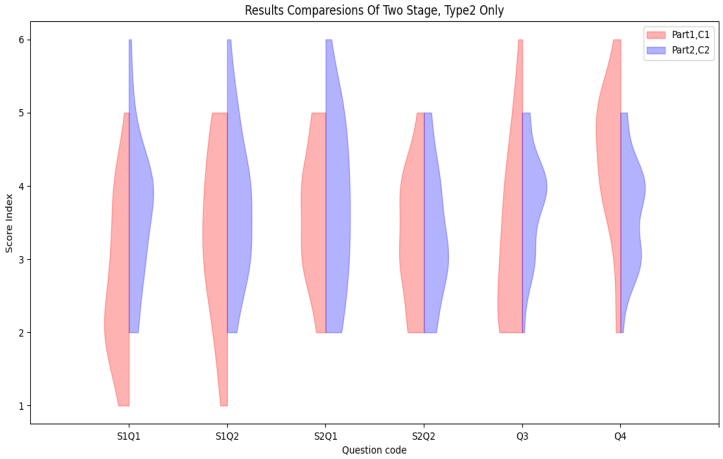
Besides, a higher variance implies disagreement among participants. Or they may encounter problems in answering. Q1 refer the music theory we do not count in the result of type 3. Although we show the results. There are still some issues that can be discussed. We provide the paired violin plot for visualizing the comparison.

The answer distribution of C3Q5 .			
	Stage1	Stage2	SUM
Type2	6	20	26
Type3	3	2	5
SUM	9	22	31

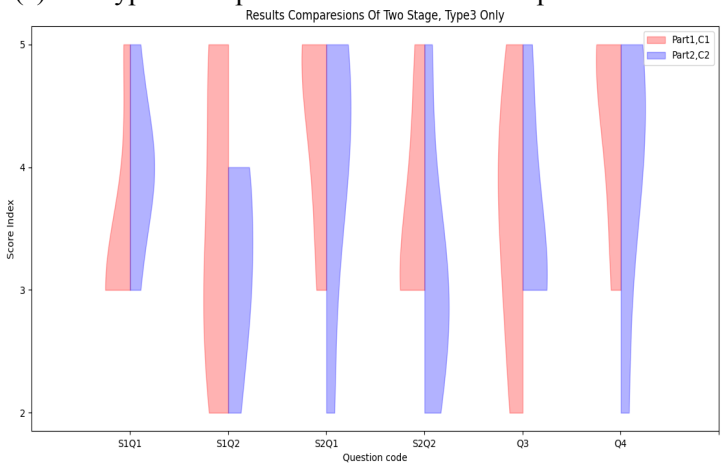
Q1 is based on the music theory, which is relatively objective. Most of the results show a lower STD with a lower degree of disagreement. While the STD of sample 1 of our model is higher, that shows more disagreement on the question of the sample. The sample 2 from the CNN-BiLSTM model has also received a higher score. But Sample 1 doesn't. The quality seems to be not as steady as our model.

Q2 and Q3 may show non-concentrated results. Because those problems are based on human feelings, this may increase the STD value. Sample 1 in the first part represents judgment. The answer to Q3 shows that the average value of part 1 is relatively low. Regardless of type 2 or type 3, STD is higher than part2. Although Q3 answered by type3 in each part get the same mean score value. This shows that our model performs well on this issue without much controversy. Answer from type 3 is more concentrated. That possibly be caused by the small sample number (only 5 of type 3). In fact, the more people who answer the question, the more non-concentrated result should be.

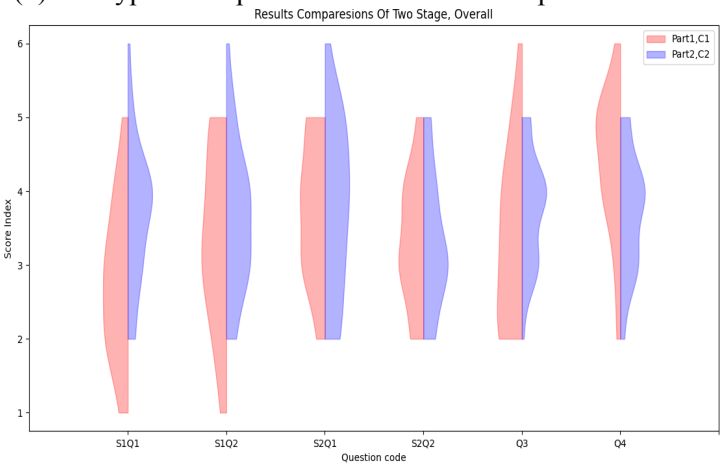
For Q4, the Turing test, the lower the score, the better result it is. The result shows our model performs well. But it does not show an obviously difference between those two models, because the distance of the two means is not large enough. Some thing interesting is observed, the type 3 gives a higher result than type 2. It's hard to explain without evidence. When observing the STD, we could find that people of type 2 reply with the answers that our model is better without doubt (STD is not too high). For type 3 people, they also think our model is better. They give the first model higher mean with a narrow score distribution as shown in Fig.4.10b



(a) The type 2 comparison results of the two parts



(b) The type 3 comparison results of the two parts



(c) The overall comparison results of the two parts

and 4.10c).

The plots shown in Figures 4.10a,4.10b, and 4.10c support our discussion and the result of Table 4.4. Overall the conclusion is that our model performs better than the CNN-BiLSTM model. Because our model gets a relative higher score on most questions except Q4 and the relative lower score on Q4 (Since the question we asked was ”these 2 samples are made by non-professionals (amateurs who have not been trained in systematic music courses, machines...)”, see previous subsections 4.4.1, and .A.). Besides, from C3Q5, our model receives more support.

Chapter 5

Conclusions And Future Work

In this thsis, we introduce the special time-series processing stream let the data more easier to train. We have designed a melody generator based on a CNN based attention model for the melody generation, combined the idea of ResNet, and pooling attention. And we add the time-distributed layer for improving the infomation matching efficiency in the attention mechanism. The deployment and training process of the model is very fast and efficient, because it has only a few parameters to be trained. So, the model does not need too much calculating resources.

Then, we use the objective evaluation system with some statistic method based on the music theory to check the model actions. The model can output melody sequences that cannot be found in sample input. Besides, The model does not rely too much on a specific few frequencies and beats, which makes the generated samples seem too monotonous and boring. The model can generate melody samples fit the basic music theory and the human aesthetics to some degrees though the subjective evaluations. This also verifies that attention model can complete the AI melody generation task.

However this is not enough. In the future, we want to combine this model with a deep reinforcement learning structure, that can add chord instrument and other music elements, for generating more complex music. Besides, we can set different behavior reward for them, which can easily realize the learning task of music style transfer for composing other styles of music.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[2] L.-C. Yang and A. Lerch, "On the evaluation of generative models in music," Neural Computing and Applications, pp. 1–12, 2018.

[3] J.-P. Briot, G. Hadjeres, and F.-D. Pachet, "Deep learning techniques for music generation," pp. 4–5, Springer, 2020.

[4] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," 2016.

[5] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "Samplernn: An unconditional end-to-end neural audio generation model," 2017.

[6] A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. van den Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis, "Parallel wavenet: Fast high-fidelity speech synthesis," 2017.

[7] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," 2018.

[8] C. Donahue, J. McAuley, and M. Puckette, "Adversarial audio synthesis," in International Conference on Learning Representations, 2019.

[9] J. Engel, K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts, "GANSynth: Adversarial neural audio synthesis," in International Conference on Learning Representations, 2019.

[10] S. Dieleman, A. van den Oord, and K. Simonyan, "The challenge of realistic music generation: modelling raw audio at scale," in Advances in Neural Information Processing Systems (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, Curran Associates, Inc., 2018.

[11] R. Manzelli, V. Thakkar, A. Siahkamari, and B. Kulis, "Conditioning deep generative raw audio models for structured automatic music," 2018.

[12] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. C. Courville, "Melgan: Generative adversarial networks for conditional

waveform synthesis," in Advances in Neural Information Processing Systems (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[13] W. Ping, K. Peng, and J. Chen, "Clarinet: Parallel wave generation in end-to-end text-to-speech," 2019.

[14] S. Kim, S. gil Lee, J. Song, J. Kim, and S. Yoon, "Flowavenet : A generative flow for raw audio," 2019.

[15] J. Serrà, S. Pascual, and C. Segura Perales, "Blow: a single-scale hyperconditioned flow for non-parallel raw-audio voice conversion," in Advances in Neural Information Processing Systems (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[16] R. Child, S. Gray, A. Radford, and I. Sutskever, "Generating long sequences with sparse transformers," 2019.

[17] M. Bińkowski, J. Donahue, S. Dieleman, A. Clark, E. Elsen, N. Casagrande, L. C. Cobo, and K. Simonyan, "High fidelity speech synthesis with adversarial networks," in International Conference on Learning Representations, 2020.

[18] W. Ping, K. Peng, K. Zhao, and Z. Song, "Waveflow: A compact flow-based model for raw audio," 2020.

[19] E. Waite, "Project magenta: Generating long-term structure in songs and stories." https://magenta.tensorflow.org/2016/07/15/lookback-rnn-attention-rnn, 2016.

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning Representations by Back-propagating Errors," Nature, vol. 323, no. 6088, pp. 533–536, 1986.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] L.-C. Yang, S.-Y. Chou, and Y.-H. Yang, "Midinet: A convolutional generative adversarial network for symbolic-domain music generation," 2017.

[23] H.-W. Dong, W.-Y. Hsiao, L.-C. Yang, and Y.-H. Yang, "Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment," 2017.

[24] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[25] C.-Z. A. Huang, T. Cooijmans, A. Roberts, A. Courville, and D. Eck, "Counterpoint by convolution," 2019.

[26] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015.

[27] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A hierarchical latent vector model for learning long-term structure in music," 2019.

[28] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," CoRR, vol. abs/1312.6114, 2014.

[29] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer," in International Conference on Learning Representations, 2019.

[30] Y.-S. Huang and Y.-H. Yang, "Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions," 2020.

[31] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014.

[32] P. Li, Y. Song, I. V. McLoughlin, W. Guo, and L.-R. Dai, "An attention pooling based representation learning method for speech emotion recognition," in Interspeech 2018, International Speech Communication Association, September 2018.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[34] "Note (i)." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000020121, 2001.

[35] W. Drabkin, "Scale." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000024691, 2001.

[36] "Whole tone." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000030241, 2001.

[37] W. Drabkin and M. Lindley, "Semitone." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000025395, 2001.

[38] "Heptatonic." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000012823, 2001.

[39] "Heptachord." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000012822, 2001.

[40] "Tonic." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000028121, 2001.

[41] W. Drabkin, "Motif." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000019221, 2001.

[42] D. Fallows, "Head-motif." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000012638, 2001.

[43] A. L. Ringer, "Melody." https://www.oxfordmusiconline.com/grovemusic/view/10.1093/gmo/9781561592630.001.0001/omo-9781561592630-e-0000018357, 2001.

[44] K.-J. Sachs and C. Dahlhaus, "Counterpoint," 2001.

[45] A. B. Downey, "Think dsp: Digital signal processing in python," ch. 8.2 Filtering and Convolution, pp. 91–93, O'Reilly Media, Inc., 1st ed., 2016.

[46] M. Hayes, Schaum's Outline of Digital Signal Processing, ch. 1.4 CONVOLUTION, pp. 11–15. Schaum's, McGraw-Hill, 1 ed., 1998.

[47] G. Blanchet and M. Charbit, Digital Signal and Image Processing using MATLAB, Volume 1: Fundamentals, ch. 4.1 Definitions and properties, pp. 115–120. Wiley-ISTE, 2 ed., 2014.

[48] G. Blanchet and M. Charbit, Digital Signal and Image Processing using MATLAB, Volume 1: Fundamentals, ch. 5.4 Frequential content of an image, pp. 198–204. Wiley-ISTE, 2 ed., 2014.

[49] M. Hayes, Schaum's Outline of Digital Signal Processing, ch. 2 Fourier Analysis, pp. 55–67. Schaum's, McGraw-Hill, 1 ed., 1998.

[50] G. Blanchet and M. Charbit, Digital Signal and Image Processing using MATLAB, Volume 1: Fundamentals, ch. 1.1.2 Spectral representation of signals, pp. 57–60. Wiley-ISTE, 2 ed., 2014.

[51] G. Blanchet and M. Charbit, Digital Signal and Image Processing using MATLAB, Volume 1: Fundamentals, ch. 2 Discrete Time Signals and Sampling, pp. 65–93. Wiley-ISTE, 2 ed., 2014.

[52] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," 2018.

[53] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," 2013.

[54] M. Lin, Q. Chen, and S. Yan, "Network in network," 2014.

[55] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in A Field Guide to Dynamical Recurrent Neural Networks (S. C. Kremer and J. F. Kolen, eds.), IEEE Press, 2001.

[56] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 157–166, 1994.

[57] J. L. Elman, "Finding structure in time," Cognitive Science, vol. 14, no. 2, pp. 179–211, 1990.

[58] P. Werbos, "Backpropagation through time: what it does and how to do it," Proceedings of the IEEE, vol. 78, no. 10, pp. 1550–1560, 1990.

[59] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," 2013.

[60] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014.

[61] H. Micko, "Attention in detection theory," in Trends in Mathematical Psychology (E. Degreef and J. Van Buggenhaut, eds.), vol. 20 of Advances in Psychology, pp. 87–103, North-Holland, 1984.

[62] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, (Lisbon, Portugal), pp. 1412–1421, Association for Computational Linguistics, Sept. 2015.

[63] "tf.keras.layers.timedistributed." https://www.tensorflow.org/api_docs/python/tf/keras/layers/TimeDistributed?hl=zh-tw.

[64] "Timedistributed layer." https://keras.io/api/layers/recurrent_layers/time_distributed/.

[65] B. McFee, V. Lostanlen, A. Metsai, M. McVicar, S. Balke, C. Thomé, C. Raffel, F. Zalkow, A. Malek, Dana, K. Lee, O. Nieto, J. Mason, D. Ellis, E. Battenberg, S. Seyfarth, R. Yamamoto, K. Choi, viktorandreevichmorozov, J. Moore, R. Bittner, S. Hidaka, Z. Wei, nullmightybofo, D. Hereñú, F.-R. Stöter, P. Friesch, A. Weiss, M. Vollrath, and T. Kim, "librosa/librosa: 0.8.0," July 2020.

[66] T. pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020.

[67] J. Brownlee, Deep Learning for Time Series Forecasting - Predict the Future with MLPs, CNNs and LSTMs in Python. 2018.

[68] D. Foster, Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play. O' Reilly Media, 2019.

[69] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[70] F. Schmidt, "Generalization in generation: A closer look at exposure bias," 2019.

[71] C. Ariza, "The interrogator as critic: The turing test and the evaluation of generative music systems," Computer Music Journal, vol. 33, no. 2, pp. 48–70, 2009.

[72] M. Pearce and G. Wiggins, "Evaluating cognitive models of musical composition," pp. 73–80, 01 2007.

[73] M. Bretan, G. Weinberg, and L. Heck, "A unit selection methodology for music generation using deep neural networks," 2016.

Appendix .A Questionnare

Title :An investigation into the rhythm of music

A wellcome page

Dear all participants, this is Unsounded Lab. This questionnaire was established by us, to helping investigate modern people's views on music melody.

Reminder: This is a questionnaire main for partial music majors. It is more suitable for people who have received music training and some professional music theory knowledge reserves (amateurs who have practiced musical instruments for more than 2 years so far can also be included). However, if you are not from the above group, we also welcome you to participate.

Thank you for participating in the survey and assisting our research topics related to music modeling. To thank you for your cooperation, we will provide prizes for a lottery.

The theoretical completion time of this questionnaire is about 6-7 minutes, divided into two parts: The first part is the precautions announcement before testing. It also includes music background checks. The second part is divided into two stages. We will place 2 melody samples in each stage. (there is no instrument added(MIDI sound only) for the consideration of only the melodies themselves. The melody fragments will not exceed 1 minute each). Two sections will be set in each stage:

Section 1: There will be 2 questions for each sample, please answer as soon as possible based on your first impression and your knowledges.

Section 2: Comprehensive questions for the sample, 2-3 questions in each stage.

Please continue if you are ready.

Section1: Precautions Announcement and The Warmming Up

Thank you for your willingness to continue to participate in this test. In order to make the results more reliable and accurate, and not to make the questionnaire conclusions misleading, please take 1-2 minutes to read the instructions and strictly follow them. I sincerely look for-

ward to and thank you for your cooperation.

Except for short answer questions and multiple-choice questions in this questionnaire, the rest of the questions are all level scale questions. Please choose according to the problem description of the topic and the current situation. 0 means the most inconsistent topic description, and 5 means the most suitable topic description:

1. Please don't do it in public/noisy places, please make sure the environment is relatively quiet before proceeding, the noise will disturb you. And please make sure that the network in your space is smooth, and there will be no disconnection or network delay. It is recommended to use a computer to complete this questionnaire, please right-click the short link in the title to open the sample in a new tab. If you want to finish the survey with your smartphone, please open the questionnaire in internet explorer(Chrome and firefox are recommend), extended press the hyperlink, and open in a new tag to get the sample.

2. Please make sure that this questionnaire is completed independently. It is best not to test with others. The actions, ideas, and unavoidable discussions of others will affect your judgment.

3. Please concentrate on this questionnaire. Before you start, please make sure that you are not in a hurry and that you are in a calm mood. You can complete it carefully within 10 minutes.

4. Please be sure to fill in the answer after the sample has been displayed.

5. Please try not to hesitate after listening, and answer the question with your first impression and music acknowledgment of music. Your delay will affect the answer and cause unexplainable conclusion deviation. It is best to complete the answer 40s-1min after the sample is broadcast.

Thank you again for your reading, understanding, and cooperation!

The formal part of the questionnaire is started.

P1-Q1: What do you want us to call you:

P1-Q2: What's your favorite style of music(can be pass):

P1-Q3: Who's your favorite musician(can be pass):

P1-Q4: Which of the following options is best for your identifications:

· Music professional background.

· Non-professional enthusiasts, who can play musical instruments of a certain degree of difficulty, and have knowledge of music theory

· None of above.

Section2

Part2 is the formal part of our questionary. This part is divided into 2 sections, 2 samples will be placed in each section. You would be asked 2 question immedeately after the display of each sample. And then after those samples are been heard, the questions are been answered, another 2 addictive question would be asked. After the 2 section finish, you would be asked to compare the sampled in the 2 different section. Click on the index title to get sample in the title. Open in a new tab. (Sample: part2-test..) Please clear you mind and stay focused. Please finish each question within 40s-1min after listening. When finished, please click one page to continue.

Explanation of nouns that will appear in the questionnaire later: structural organization, which refers to the similarity of pitch sequence in the evolution of melody, (for example, Do-Mi-So-Fa, Re-Fa-La-So are arranged in pitches in music. Similar, doing -Mi-So-Fa itself is counted as repeating before and after. The beat bar of a melody will be repeated. The pitch arrangement and the repetition and similarity of the beat are considered by the structural organization.

Please Be sure to Read the explaination.

Part1-Stage1

Sample1 · C1S1Q1: Do you agree that the sample melody is structurally organized (beats, pitch sequence structure, mainly pitch), and the melody changes steadily and smoothly (the pitch difference between before and after the melody is not more than 8 degrees during the evolution of the melody). Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

· C1S1Q2: Do you think the sample has affected your emotions at the moment (pleasure, comfort, or melancholy, sadness,...), does it remind you of anything (memories, stories, other music, familiar people, exotic feelings, memories) , Stories, other music, familiar people, exotic customs...) .Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

Sample2 · C1S2Q1: Do you agree that the sample melody is structurally organized (beats, pitch sequence structure, mainly pitch), and the melody changes steadily and smoothly (the pitch difference between before and after the melody is not more than 8 degrees during the evolution of the melody). Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

· C1S2Q2: Do you think the sample has affected your emotions at the moment (pleasure, comfort, or melancholy, sadness,...), does it remind you of anything (memories, stories, other

music, familiar people, exotic feelings, memories) , Stories, other music, familiar people, exotic customs...). Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

Stage1 finish.

Part1-Stage2 For the two samples in the first stage just now, answer the following two questions :

· C1Q3: Take the 2 samples in this stage together, the styles of them are different, and they don't sound monotonous, and they won't make you feel unpleasant (piercing, feeling bad or strange, inconsistent, inconsistent).Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

· C1Q4: Take the 2 samples in this stage together, these 2 samples are made by non-

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

professionals (amateurs who have not been trained in systematic music courses, machines...). Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

Part 1 is finish. Now goes to Part 2.

Part2-Stage1

Sample1 · C2S1Q1: Do you agree that the sample melody is structurally organized (beats, pitch sequence structure, mainly pitch), and the melody changes steadily and smoothly (the pitch difference between before and after the melody is not more than 8 degrees during the evolution of the melody). Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

· C2S1Q2: Do you think the sample has affected your emotions at the moment (pleasure, comfort, or melancholy, sadness,...), does it remind you of anything (memories, stories, other music, familiar people, exotic feelings, memories) , Stories, other music, familiar people, exotic customs...) .Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

Sample2 · C2S2Q1: Do you agree that the sample melody is structurally organized (beats, pitch sequence structure, mainly pitch), and the melody changes steadily and smoothly (the pitch difference between before and after the melody is not more than 8 degrees during the evolution of the melody). Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

· C2S2Q2: Do you think the sample has affected your emotions at the moment (pleasure, comfort, or melancholy, sadness,...), does it remind you of anything (memories, stories, other music, familiar people, exotic feelings, memories) , Stories, other music, familiar people, exotic customs...). Choose the option below that you think best fits the description of the topic, and

the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

Stage1 finish .

Part2-Stage2 For the two samples in the first stage just now, answer the following two questions :

· C2Q3: Take the 2 samples in this stage together, the styles of them are different, and they don’t sound monotonous, and they won’t make you feel unpleasant (piercing, feeling bad or strange, inconsistent, inconsistent).Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

· C2Q4: Take the 2 samples in this stage together, these 2 samples are made by non-professionals (amateurs who have not been trained in systematic music courses, machines...). Choose the option below that you think best fits the description of the topic, and the option is your rating for this topic.

	1.Totally Disagree	2.Disagree	3.Slightly Disagree	4.Slightly Agree	5.Agree	6.Totally Agree
Scales						

Stage 2 is finish.

Now, it is the last question:

· C3Q5: The which two samples in which stage do you prefer?

· The Stage 1.

· The Stage 2.