Title

Controllable Monophonic Music Generation via Latent Variable Disentanglement

Author

Chen, Ke

Publication Date

2021

UNIVERSITY OF CALIFORNIA SAN DIEGO

Controllable Monophonic Music Generation via Latent Variable Disentanglement

A thesis submitted in partial satisfaction of the
requirements for the degree Master of Arts

in

Music

by

Ke Chen

Committee in charge:

Professor Shlomo Dubnov, Chair
Professor Taylor Berg-Kirkpatrick
Professor Miller Puckette

2021

The Thesis of Ke Chen is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank Professor Shlomo Dubnov, Professor Taylor Berg-Kirkpatrick and Professor Miller Puckette for their support as the members of my committee. Through multiple drafts and many long nights, their guidance has proved to be invaluable. Their wisdom inspired me and allowed me to find solutions when I encountered research bottlenecks. Additional thanks to countless peers in the UCSD Department of Music graduate program for support in daily affairs and communication.

I would also like to thank Professor Gus Xia in Music X Lab at New York University Shanghai. Without him, my research on music generation would not have started. It is his generous availability and support that helped me get into this field.

Finally, I would like to thank my parents Xiumei Li and Xiaozhou Chen for their love. They always support my decision, no matter whether there are unknown difficulties ahead.

Chapter 1, Chapter 3, and Chapter 6 contain some materials (texts, tables, and figures) from a published conference paper: Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick and Shlomo Dubnov, *Music SketchNet: Controllable Music Generation via Factorized Representations of Pitch and Rhythm*, in Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020. The thesis author was the first author of this paper.

Chapter 2, Chapter 4, and Chapter 5, in part, are currently being prepared for submission for publication of the material. Ke Chen; Taylor Berg-Kirkpatrick; Shlomo Dubnov. The thesis author will be the first author of this paper.

ABSTRACT OF THE THESIS

Controllable Monophonic Music Generation via Latent Variable Disentanglement

by

Ke Chen

Master of Arts in Music

University of California San Diego, 2021

Professor Shlomo Dubnov, Chair

Automatic music generation is an attractive topic in the interdisciplinary field of music and computer science. The appearance of deep learning technique has brought in new methodologies to this topic. Diving to this topic inspires us to understand how computers process music elements from notes, beats to melodies, structures and dynamics. This further helps humans to better understand the music if we could afterwards extract creation mechanisms from machines.

In the generation problem, how to make human interact with the computer is an interesting problem. Drawing an analogy with automatic image completion systems, we propose Music SketchNet, a neural network framework that allows users to specify partial musical ideas guiding monophonic music generation. We focus on generating the missing measures in incomplete monophonic musical pieces, conditioned on surrounding context, and optionally guided by user-specified pitch and rhythm snippets.

First, we introduce SketchVAE, a novel variational autoencoder that explicitly factorizes rhythm and pitch contour to form the basis of our proposed model. Then we introduce two discriminative architectures, SketchInpainter and SketchConnector, that in conjunction perform the guided music completion, filling in representations for the missing measures conditioned on surrounding context and user-specified snippets. In the experiment, we first evaluate the SketchVAE on three standard datasets from different genres including folk, classic and pop songs. Then we evaluate the whole SketchNet on a standard dataset of Irish folk music and compare with models from recent works. When used for music completion, our approach outperforms the state-of-the-art both in terms of objective metrics and subjective listening tests. Finally, we demonstrate that our model can successfully incorporate user-specified snippets during the generation process.

Chapter 1

Introduction

1.1 Introduction

As a research area, automatic music generation has a long history of studying and expanding human expression/creativity [30]. The use of neural network techniques in automatic music generation tasks has shown promising results in recent years [7]. In this paper, we focus on a specific facet of the automatic music generation problem on how to allow users to flexibly and intuitively control the outcome of automatic music generation.

Inspired by the sketching and patching work from computer vision [4, 21, 35, 43, 40], we propose Music SketchNet[1] which allows users to specify partial musical ideas in terms of incomplete and distinct pitch and rhythm representations. More specifically, we generalize the concept of sketching and patching – wherein a user roughly sketches content for a missing portion of an image – to music, as depicted in Figure 1.1. There are two contexts: previous context and future context as the condition fed into the proposed model. The missing music measures in the middle is the target that we train a model to fill in. Besides making the completion of missing measures, we allow users to specify two music ideas: the rhythm pattern and the pitch contour during the generative process. These specifications are taken into account by the model when generating the missing measures. As a result, the proposed framework will complete the missing parts given the known context and user input. More importantly, the entire generation process is

[1] https://github.com/RetroCirce/Music-SketchNet.

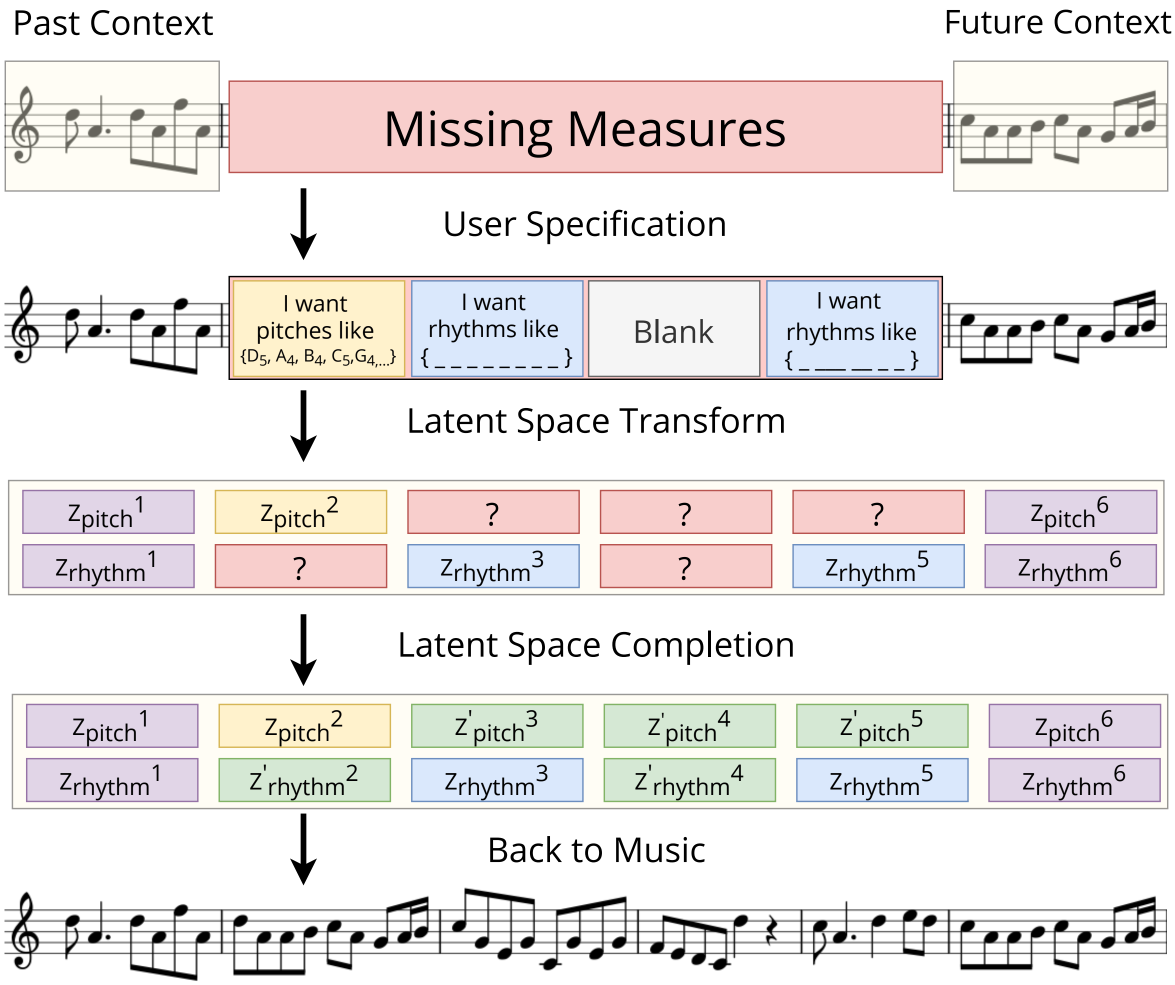

Figure 1.1. The music sketch scenario. The model is designed to fill the missing part based on the known context and user's own specification.

carried out in the latent space. The model does not gradually generate the music point by point, or step by step. Instead, each time it generates a music vector in a high-dimensional latent space, and each vector represents a music measure after decoding. Therefore, this generative process of the model is closer to the creation process of musicians (sentence-by-sentence vs. note-by-note).

1.2 Novel Contribution

This thesis introduces a new framework Music SketchNet, that allows users to specify partial musical ideas in terms of incomplete and distinct pitch and rhythm patterns in the automatic music generation. This interactive model is taken as a great example to explore the possibility of machines to create music with human beings. To work through all these procedures, Music SketchNet consists of three components as our listed contributions:

- SketchVAE, a variational autoencoder (VAE) [29] component for mapping monophonic music measures into high-dimensional latent vectors. By the use of a factorized inference network, SketchVAE decouples latent variables into two parts: pitch contour and rhythm, which serve as the control parameters for users.

- SketchInpainter, a component for generating music given the previous and future contexts via recurrent neural network (RNN) [33]. It contains stacked recurrent networks to handle the element-level inpainting prediction in the latent space.

- SketchConnector, a transformer-based [38] component for handling user's control of musical pitch and rhythm patterns. It receives users' sketches of pitch, rhythm, or both, combines them with the prediction from SketchInpainter, and finalizes the generation.

- Three components form a novel model to let users specify their ideas in monophonic music generation.

In this thesis, we show that the proposed SketchVAE is capable of factorizing music input into latent variables meaningfully. The proposed SketchInpainter and SketchConnector allow users to control the generative process. The novel training and evaluation methodologies of the SketchConnector are also presented.

1.3 Thesis Organization

This thesis will begin in Chapter 2, by discussing several conditional music generation models as the related works, and some machine learning backgrounds of our proposed model, including recurrent neural network, variational autoencoder, and attention network. The proposed model Music SketchNet will be introduced in detail in Chapter 3, including its data processing, three components, and training paradigm. In Chapter 4, four music datasets will be covered. Among them, three datasets (Nottingham, Hynmal, Wikifornia) will be used in the SketchVAE to evaluate the performance of representation accuracy and latent space capacity in Chapter 5. The left Irish and Scottish Folk Song dataset will be used in the whole SketchNet model to evaluate the generation performance in Chapter 6. Some discussions and conclusions will appear in Chapter 7.

This chapter contains some materials (texts, tables, and figures) from a published conference paper: Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick and Shlomo Dubnov, Music SketchNet: Controllable Music Generation via Factorized Representations of Pitch and Rhythm, in Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020. The thesis author was the first author of this paper.

Chapter 2

Background and Related Work

In this chapter, we introduce three important structures of deep learning that relevant to our proposed model. Then we go over some related works on conditional music generation, disentanglement methodologies, and discuss the further progress our model makes based on them.

2.1 Recurrent Neural Network

A recurrent neural network (RNN) is a class of artificial neural network. It is first proposed by David Rumelhart and further discovered by John Hopfield [33]. Different from the vanilla fully-connected neural network, RNN contains a internal hidden state that stores the information from the previous time steps' input. Therefore, RNN is designed to capture some temporal relations along with the given sequence. It has been widely applied into the practical scenarios that have time series data (e.g. text generation, image generation, image caption, and music generation).

As depicted in Figure 2.1 [12], a recurrent neural network contains an input vector x_t, an internal hidden state h_t, and an output vector o_t. The subscript t refers to the time step (i.e. for each time step, the network will have different input, hidden state and output). And V, U, W refer different weight matrices. There are two different calculation method of these three vectors: An Elman network [16] and a Jordan network [28]. The Elman network is more widely used in

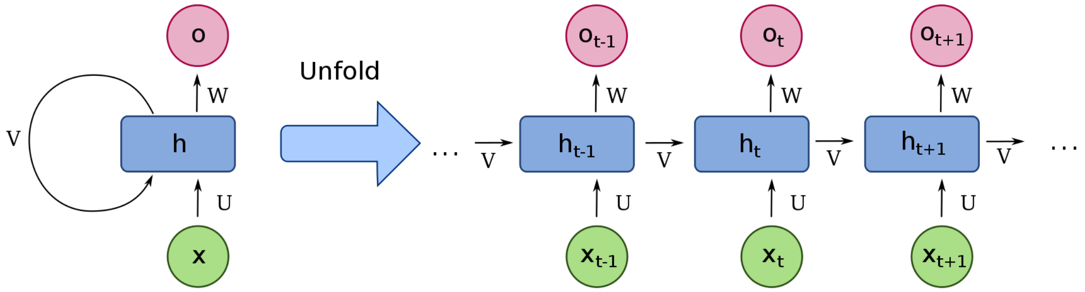

Figure 2.1. The recurrent neural network architecture, referred from [12]

current works:

$$h_t = \sigma_h(Ux_t + Vh_{t-1} + b_h) \tag{2.1}$$

$$o_t = \sigma_o(Wh_t + b_o) \tag{2.2}$$

Where σ_h, σ_o are activation functions of the hidden state and the output vector, b_h, b_o are bias weights. Similar to any artificial neural network, we adopt back-propagation throught time (BPTT) [39] in update the weights with given training sequential data.

However, the major problem of recurrent neural network is the long-term dependency. The hidden state h is updated in every time step within the input. Intuitively, the hidden state stores too much information from past to future and cannot forget anything. And this is inconsistent with how human deals with the memory (i.e. retention and forget). Mathematically, the hidden state has accumulated a large number of gradients in the recurrent loop. When it is being updated, the gradient will vanish or explode. This brings great difficulties to the update of the model. Therefore, RNNs are very difficult to train.

To address this problem, a redesign of the hidden state has been proposed. Two famous structure appear: long short-term memory (LSTM) [25] and gated recurrent unit (GRU) [11]. In this thesis, we use GRU, which is a more efficient network than LSTM.

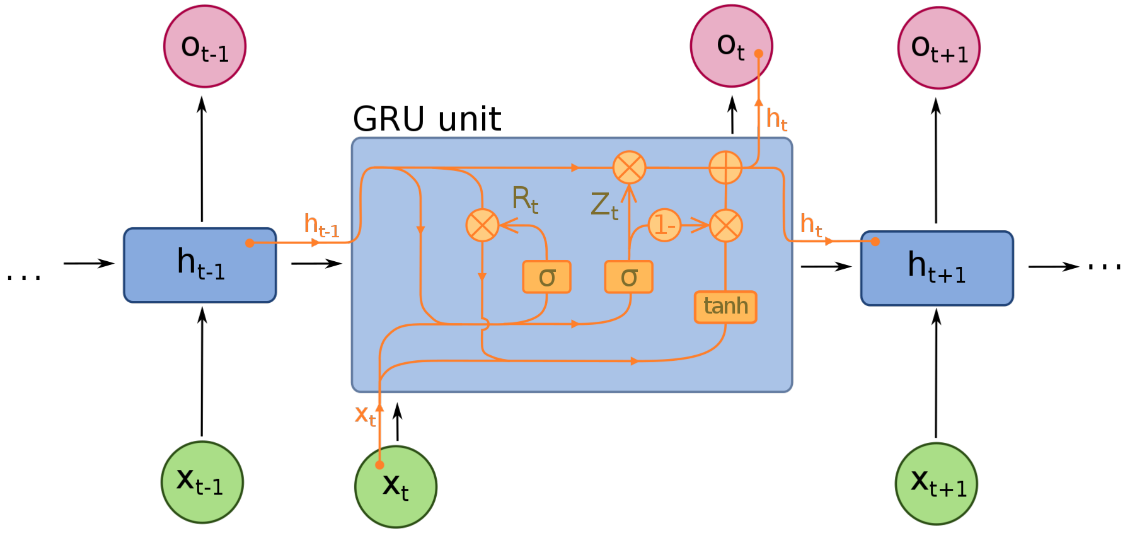

Figure 2.2. The gated recurrent unit architecture, referred from [12]

A gated recurrent unit contains several gates to better help the network capture, utilize and forget the memory. The calculation of each vector is shown below:

$$z_t = \sigma_g(U_z x_t + V_z h_{t-1} + b_z) \tag{2.3}$$

$$r_t = \sigma_g(U_r x_t + V_r h_{t-1} + b_r) \tag{2.4}$$

$$c_t = tanh(U_c x_y + V_c(r_t \odot h_{t-1}) + b_c) \tag{2.5}$$

$$h_t = (1 - z_t)h_{t-1} + z_t \odot c_t \tag{2.6}$$

Where σ_g is the Sigmoid function, z_t, r_t are two gate 0-1 values that determine if the current cell state needs to preserve the last hidden state, and how much the current hidden state needs to preserve the current cell state.

With this design, GRU could determine the state of the memory by controlling the gate in limiting the communication of cell and hidden states. It not only mitigates the gradient vanishing problem of RNN, but also makes the network move further into the human cognition of memory.

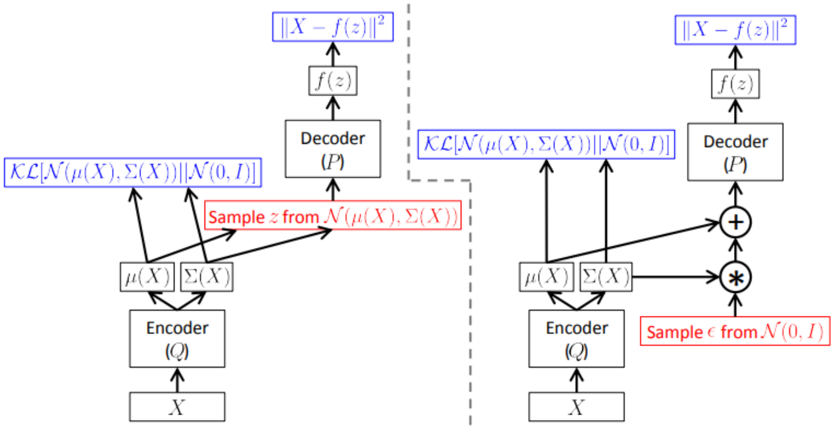
$\|X - f(z)\|^2$
$f(z)$
Decoder (P)
$\mathcal{KL}[\mathcal{N}(\mu(X), \Sigma(X))\|\mathcal{N}(0, I)]$
Sample z from $\mathcal{N}(\mu(X), \Sigma(X))$
$\mu(X)$
$\Sigma(X)$
Encoder (Q)
X
$\|X - f(z)\|^2$
$f(z)$
$\mathcal{KL}[\mathcal{N}(\mu(X), \Sigma(X))\|\mathcal{N}(0, I)]$
Decoder (P)
+
*
$\mu(X)$
$\Sigma(X)$
Encoder (Q)
Sample ϵ from $\mathcal{N}(0, I)$
X

Figure 2.3. A training-time variational autoencoder [14] implemented as a feedforward neural network, where $p(x|z)$ is Gaussian. Left is without the "reparameterization trick", and right is with it. Red shows sampling operations that are non-differentiable. Blue shows loss layers.

2.2 Variational Autoencoder

A normal autoencoder is a deep learning structure that contains an encoder and a decoder. Both encoder and decoder can compose of any type of neural network. During the training process, an input is encoded into a latent vector, and an output is then decoded from the same latent vector. The goal is to make the input and output as similar as possible. An autoencoder is usually used as a dimensional reduction model, or a representation model. It could map a complex data into a high-dimensional latent vector while maintaining most of its information. Moreover, these latent vectors can be used in many down-streaming tasks because they themselves are another advanced form of data.

On top of normal Auto-Encoders, variational autoencoders (VAE) explicitly constrains that the latent variable z should be a random variable distributed according to a prior $p(z)$. The input x and latent code z can then be seen as $z \sim p(z), x \sim p(x|z)$. The VAE consists of an encoder $q_\lambda(z|x)$, which approximates the posterior $p(z|x)$, and a decoder $p_\theta(x|z)$, which parameterizes the likelihood $p(x|z)$. The approximate posterior and likelihood distributions are parameterized by neural networks. Posterior inference is done by minimizing the KL divergence between the encoder and the true posterior. It can be proven that this optimization problem is the same as maximizing the evidence lower bound(ELBO):

$$ELBO = E[\log p_\theta(x|z)] - KL(q_\lambda(z|x)||p(z)) \leq \log p(x) \tag{2.7}$$

In practice, computing the gradient through ELBO is infeasible due to the sampling of z. A common approach is to assume that $p(z)$ is a diagonal-covariance Gaussian $z \sim \mathcal{N}(\mu, \sigma)$. Figure 2.3 [14] shows a variational autoencoder implemented as a feedforward neural network.

2.3 Attention Network

Attention mechanism is proposed in [38] by Google Brain. In traditional recurrent neural network, the model can only learn the relation from past to future, or reversely with bidirectional design. However, in many scenarios, the prediction or generation of a certain time step value not only depends on the past information, but the contributions of the information at past time steps are also different. For example, in the text generation, the currently generated word sometimes largely depends on nouns in the preamble, rather than verbs (or vice versa).

To address this problem, similar to RNN, an attention network is settled in a sequential problem where the input embedding of each time step is x_t. Different from devising the hidden state h_t in RNN, an attention network devises three variables: Q (query), K (key), and V (value). In self-attention network, all of them are obtained by a linear transform of the data:

$$Q = W_q X + B_q \tag{2.8}$$

$$K = W_k X + B_k \tag{2.9}$$

$$V = W_v X + B_v \tag{2.10}$$

Where X is a vector matrix from x_1 to x_T (the maximum time step). Q, K, V are therefore obtained as three matrices, each of which contains vectors from the 1st time step to the final T-th time step. All Q, K, V are in the same size.

Then the attention output is obtained by:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.11}$$

Where d_k is the size of Q, K, V as a normalized denominator. Mathematically, the attention output shows that each time step attention output is the result of multiplying all time step values V with different proportions. And this proportion is calculated from the query value at the current time q_t and the key value at all times K.

Therefore, attention mechanism can calculate the contributions (or attention scores) of inputs in different time steps to the current time step input. And this network is proved to have a higher efficiency and better performance than RNNs in many related tasks.

2.4 Conditional Music Generation

Prior work supports various forms of conditional music generation, depending on the condition form and the generation target. MuseGan [15] is trained by generative adversarial network (GAN) [19] and allows users to condition generated results on full-length multi-track music. DeepBach [23] provides a constraint mechanism that allows users to limit the generated results to match composer styles (i.e. Johann Sebastian Bach's style). Music Transformer [26] supports a accompaniment arrangement from an existing melody track in classical music via the latest attention mechanism in deep learning field. However, all these approaches require the user preference to be defined in terms of complete musical tracks. Different from above works, our Music SketchNet proposes a new condition scenario called sketching: given the known contexts and user controls of some musical elements, a model should make the completion of the missing music while reflecting the user's specification. In this thesis, the known contexts are defined as the previous and future contexts, and the user controls are defined as two factors: pitch contour and rhythm pattern.

There has been limited work on sketching in music generation. Some work [34, 23] has used Markov Chain Monte Carlo (MCMC) to generate music with given contexts or generate music conditioned on simple starting and ending notes [22]. The most related task is music inpainting: completing a musical piece by generating a sequence of missing measures given the surrounding context, but without conditioning on any form of user preferences. Music InpaintNet [31] completes musical pieces by predicting vector representations for missing measures, then the vector representations are decoded to output symbolic music through the use of a variational autoencoder.

Our proposed music sketching scenario takes music inpainting a step further. We let users specify musical ideas by controlling pitch contours or rhythm patterns, not by complete musical tracks. The user input is optional: users can choose to specify musical ideas, or let the system fill in predictions without conditioning on user preferences.

2.5 Disentanglement Learning

One of the key problems for our proposed Music SketchNet is how to disentangle the music elements (e.g. rhythm and pitch) in the latent space. In the Literature, there are several works attempting to disentangle elements in other field. In the computer vision, [17] divides the loss of image generation into the content loss and the style loss to realize the style transfer from one image to another image. In the language processing, [36] factorizes the font into the character-specific content and the font-style content to change the font-style of texts. In the computer music field, the most related work is EC^2-VAE [41].

EC^2-VAE factorizes music measures with separate vectors representing pitch and rhythm by using a universal encoder, a separate rhythm decoder, and a universal decoder. The latent variables can then be divided in half between pitch contour and rhythm. Exchanging different halves of the disentangled latent dimensions can be thought of as ”analogy generation”, where song A inherits the original pitch contour but has the rhythm of song B. EC^2-VAE can perform the same function – factorizing the pitch and rhythm, as SketchVAE can perform. The difference between EC^2-VAE and our SketchVAE is that EC^2-VAE contains one encoder and two decoders, while SketchVAE contains two encoders and one decoder. In that, they are particularly worthy of comparison structurally in this thesis.

This chapter, in part, are currently being prepared for submission for publication of the material. Ke Chen; Taylor Berg-Kirkpatrick; Shlomo Dubnov. The thesis author will be the first author of this paper.

Chapter 3

Music SketchNet

We formalize the music sketching task as solving the following three problems: (1) how to represent music ideas or elements, (2) how to generate new materials given the past and future musical context and (3) how to process users' input and integrate it with the system. A visualization of the sketching scenario is depicted in Figure 1.1.

We propose three neural network components to tackle the three problems. The SketchVAE encodes/decodes the music between external music measures and the learned factorized latent representations. The SketchInpainter predicts musical ideas in the form of the latent variables given known context. And the SketchConnector combines the predictions from SketchInpainter and users' sketching to generate the final latent variables which are sent into the SketchVAE decoder to generate music output. A diagram showing the proposed pipeline is shown in Figure 3.1.

3.1 Problem Definition

More formally, the proposed sketch framework can be described as a joint probability model of the missing musical content, X^m, conditioned on the past, future, and user sketching input. The joint probability breaks down into a product of conditional probabilities corresponding

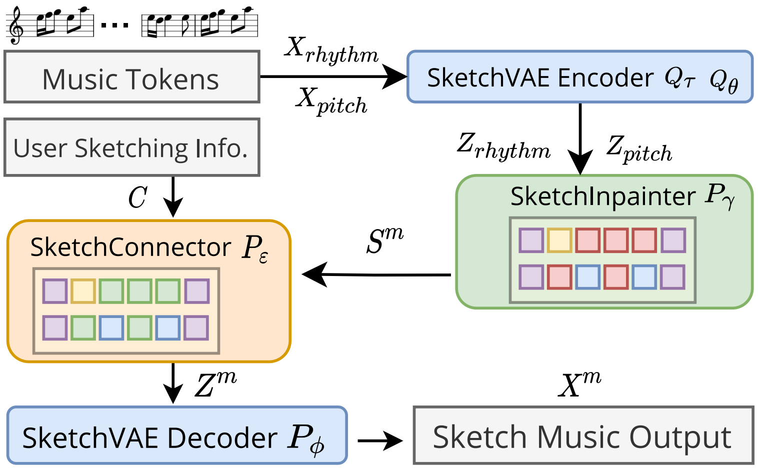
Music Tokens
X_{rhythm}
X_{pitch}
SketchVAE Encoder Q_τ Q_θ
User Sketching Info.
C
Z_{rhythm}
Z_{pitch}
SketchInpainter P_γ
S^m
SketchConnector P_ε
Z^m
X^m
SketchVAE Decoder P_ϕ
Sketch Music Output

Figure 3.1. The Music SketchNet pipeline. The color patterns inside Inpainter and Connector correspond to the latent space transform and completion process in Figure 1.1.

to sub-components of the framework:

$$P_{\phi,\varepsilon,\gamma,\theta,\tau}(X^m,Z,S|X^p,X^f,C) = \tag{3.1}$$

$$P_\phi(X^m|Z^m) \tag{SketchVAE Decoder}$$

$$*P_\varepsilon(Z^m|S^m,C) \tag{SketchConnector}$$

$$*P_\gamma(S^m_{pitch}|Z^p_{pitch},Z^f_{pitch}) \tag{SketchInpainter}$$

$$*P_\gamma(S^m_{rhythm}|Z^p_{rhythm},Z^f_{rhythm}) \tag{SketchInpainter}$$

$$*Q_\theta(Z^p_{pitch},Z^f_{pitch}|X^p_{pitch},X^f_{pitch}) \tag{3.2}$$

$$*Q_\tau(Z^p_{rhythm},Z^f_{rhythm}|X^p_{rhythm},X^f_{rhythm}) \tag{SketchVAE Encoders}$$

X indicates the input/output music sequence, Z is the sequence for $\{z\}$ the latent variable , S is the SketchInpainter's predicted sequence, C is users' sketching input. The superscripts, p, m, f indicate the past, missing and future context. The subscripts, $pitch$ and $rhythm$ indicate the pitch and rhythm latent variables. Q_θ, Q_τ, P_ϕ are the SketchVAE pitch/rhythm encoders and decoder parameters, P_γ represents the SketchInpainter, and P_ε is the SketchConnector.

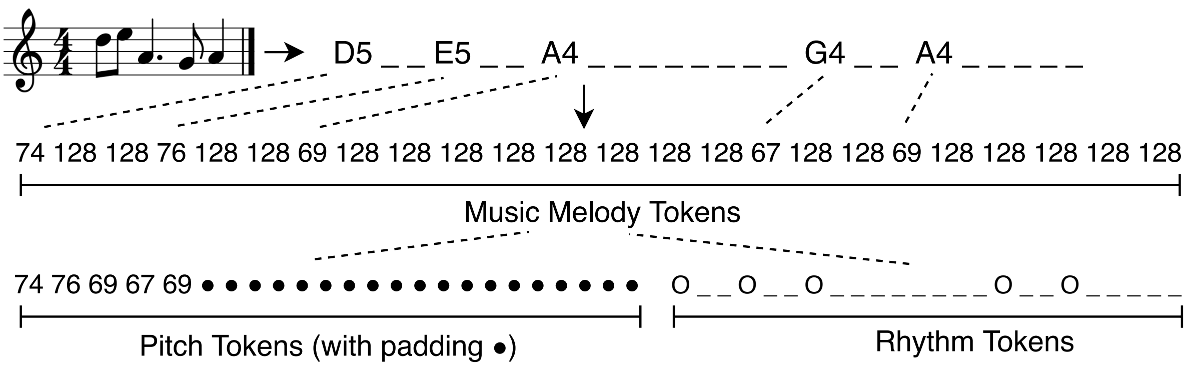

Figure 3.2. An example of the encoding of a monophonic melody.

3.2 SketchVAE for Representation

MusicVAE [32] is one of the first works applying the variational auto-encoder [29] to music. MeasureVAE [31] further focuses on representing isolated measures and utilizes a hierarchical decoder to handle ticks and beats. EC^2-VAE [41] factorizes music measures with separate vectors representing pitch and rhythm by a single encoder and two decoders. Our proposed SketchVAE aims to factorize representations by introducing a factorized encoder that considers pitch and rhythm information separately in the encoder channels. Different from EC^2-VAE, it could allow users to enter parts of the information (rhythm and/or pitch) optionally.

SketchVAE aims to represent a single music measure as a latent variable z that encodes rhythm and pitch contour information in separate dimensions (z_{pitch}, z_{rhythm}). It contains (1) a pitch encoder $Q_\theta(z_{pitch}|x_{pitch})$, (2) a rhythm encoder $Q_\tau(z_{rhythm}|x_{rhythm})$, and (3) a hierarchical decoder $P_\phi(x|z_{pitch}, z_{rhythm})$ as shown in Figure 3.3.

3.2.1 Music Score Encoding

Similar to [32], we encode the monophonic midi melody by using [0, 127] for the note onsets, 128 for holding state, and 129 for the rest state. We cut each measure into 24 frames to correctly quantize eighth-note triplets like [31], and encode the midi as described in the previous sentence.

As Figure 3.2 shows, we further process the encoded 24-frame sequence x into x_{pitch} and

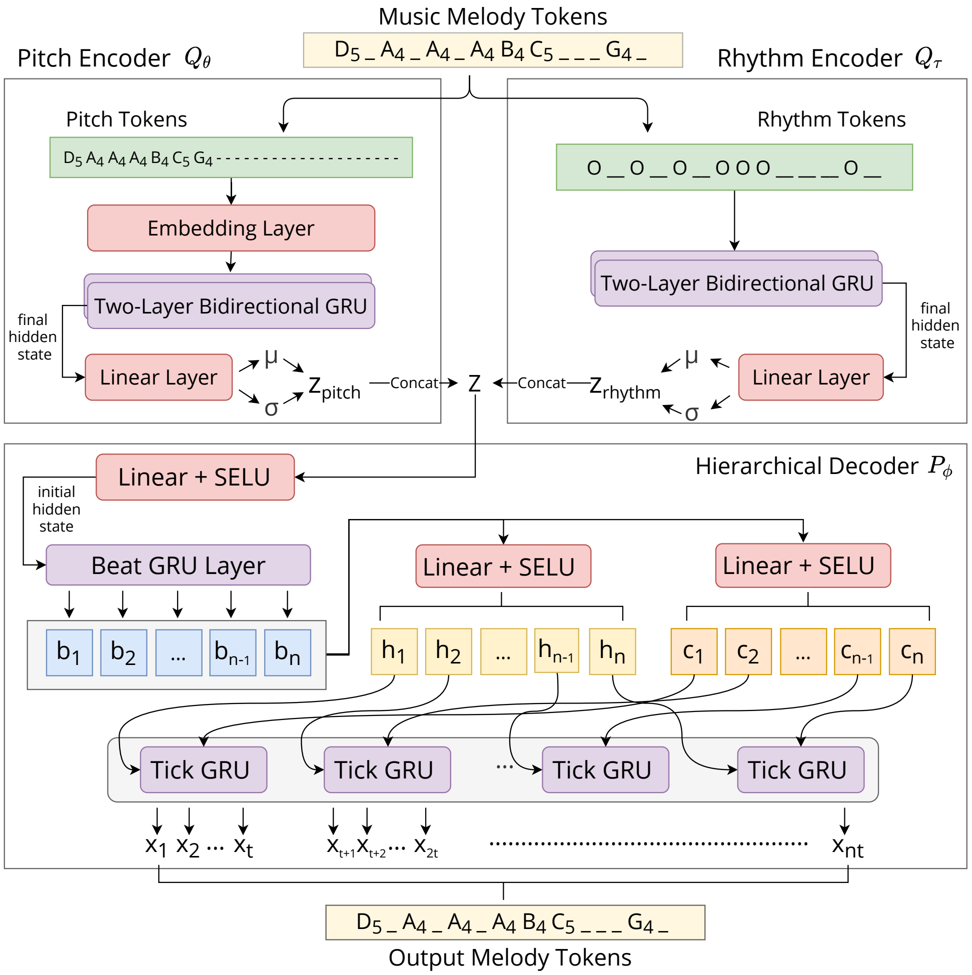

Figure 3.3. SketchVAE structure: pitch encoder, rhythm encoder and hierarchical decoder. Rhythm tokens: the upper dashes denote the onsets of note, and the bottom dashes denote the hold/duration state. We use pitch symbols to represent the tokens numbers for better illustration.

x_{rhythm}, the pitch and rhythm token sequences respectively. The pitch token sequence x_{pitch} is obtained by picking all note onsets in x with padding (shown by ”•” in Figure 3.2) to fill 24 frames. The rhythm token sequence x_{rhythm} is obtained by replacing all pitch onsets with the same token (shown by ”O” and ”_” in Figure 3.2). A similar splitting strategy is also used in [18]. Our motivation is to provide users with two intuitive music dimensions to control, and to help enforce better factorization in the latent representation for later prediction and control.

3.2.2 The Pitch Encoder and Rhythm Encoder

After pre-processing x, x_{pitch} only contains the note value sequence, while x_{rhythm} only has the duration and onset information. x_{pitch} and x_{rhythm} are then fed into two different GRU [11] encoders for variational approximation. The outputs of each encoder are concatenated into $z = [z_{pitch}, z_{rhythm}]$.

3.2.3 The Hierarchical Decoder

After we obtain the latent variable z, we feed it into the hierarchical decoder. This decoder is similar to the decoder used in MeasureVAE [31]. As shown in the bottom part in Figure 3.3, it contains an upper ”beat” GRU layer and a lower ”tick” GRU layer. This division’s motivation is to decode z into n beats first and then decode each beat into t ticks. As a result, the note information in each measure will be decoded in a musically intuitive way. For the tick GRU, we use the teacher forcing [5, 20] and auto-regressive techniques to train the network efficiently. The output is conditioned frame-by-frame not only on the beat token but also on the last tick token.

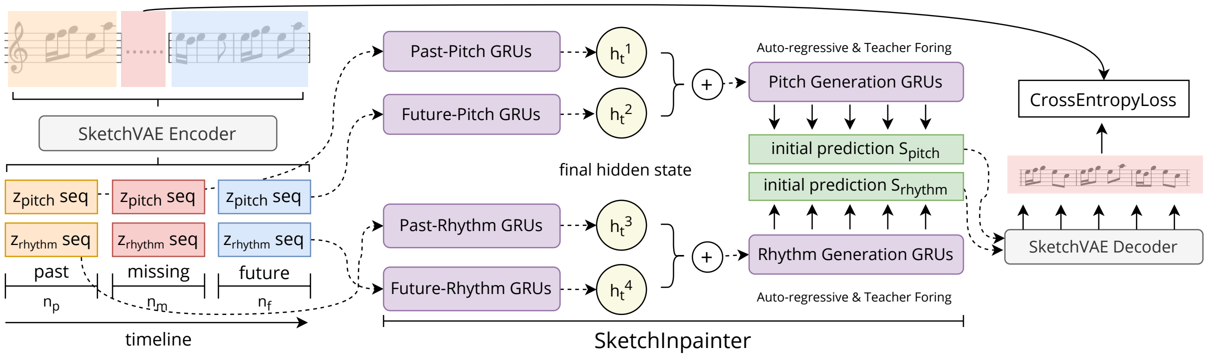

Figure 3.4. SketchInpainter structure. We feed the music tokens into the SketchVAE and obtain the latent variable sequences. And we feed the sequences into the pitch GRU and the rhythm GRU groups to generate the initial prediction S.

3.2.4 Encoding the Past, Missing and Future Musical Context

The latent variable sequences Z^p, Z^m, and Z^f are then obtained by processing the music input in measure sequences X^p, X^m, and X^f. Both X^m and Z^m are masked during training. This encoding part is shown in the left block of Figure 3.4.

3.3 SketchInpainter for Initial Prediction

Next, we describe the model component that performs the music inpainting to predict latent representations for the missing measures. The SketchInpainter accepts Z_{pitch} and Z_{rhythm} as two independent inputs from SketchVAE. Then only the past and future Z_{pitch} and Z_{rhythm} are fed into the pitch/rhythm GRU groups respectively. The output from each GRU group is the hidden state h, as shown in the middle of Figure 3.4.

Then we combine the past/future hidden states h from both the pitch and rhythm GRU groups and use them as the initial states for the pitch/rhythm generation GRUs. The generation GRUs then predict the missing latent variables by $S^m = (S_{pitch}, S_{rhythm})$, as shown in the green box in Figure 3.4. Each generation GRU is trained with the teach forcing and auto-regressive techniques.

Each output vector s^m from S^m has the same dimension as the latent variable z from Z. We first build a model with only SketchVAE and SketchInpainter that directly predicts the missing music material, X^m. As the right block of Figure 3.4 shows, S^m is sent into the SketchVAE decoder and we compute the cross entropy loss between the predicted music output and the ground truth. This is the stage I training in our model, detailed in Section 3.3.

3.4 SketchConnector for Finalization

The predicted S^m from SketchInpainter can already serve as a good latent representation for the missing part X^m. We continue by devising the SketchConnector, $P_\varepsilon(Z^m|S^m, C)$, to modify the prediction with user control. To make up for the lack of correlation between pitch and

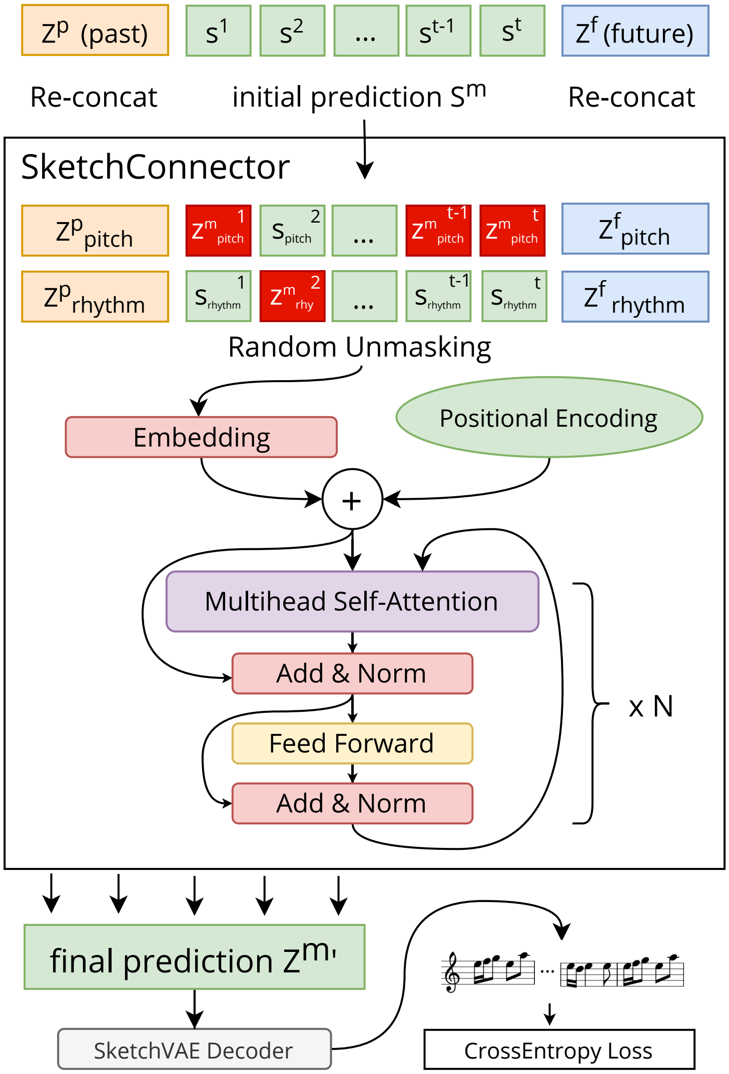

Figure 3.5. The SketchConnector: the output of SketchInpainter is randomly unmasked and fed into a transformer encoder to get the final output.

rhythm in current predictions, we introduce the SketchConnector as a way to intervene/control the generative process, that also leads to a wider musical expressivity of the proposed system.

3.4.1 Random Unmasking

With S^m obtained from SketchInpainter, we concatenate it with Z^p and Z^f again. However, before we feed it back into the network, we randomly unmask some of the missing parts to be the ground-truth (simulating user providing partial musical context). The masked S^m are shown by the red boxes in Figure 3.5. We replace some s from S^m to be the real answer in Z^m, denoted as

C. We observe that this optimization is very similar to BERT [13] training. The difference is that BERT randomly masks the ground truth labels to be unknown, but SketchConnector randomly unmasks the predictions to be truths. The unmasking rate is set to 0.3.

Intuitively, this allows the model to learn a more close relation among current rhythm, pitch tokens, and the nearest neighbour tokens. In the sketch inference scenario, the randomly unmasked measures will be replaced by the user sketching information, which allows a natural transition between the training and testing process.

3.4.2 Transformer-based Connector

Then with S^m and the random unmasking data C, we feed them into a transformer encoder with absolute positional encoding. In contrast to [26], we do not use relative positional encoding because our inputs are vectors representing individual measures, whose length is far shorter than midi-event sequences.

The output of the SketchConnector, Z^m, will be the final prediction for the missing part. We feed it into the SketchVAE decoder, and compute the cross entropy loss of the output with the ground-truth.

This chapter contains some materials (texts, tables, and figures) from a published conference paper: Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick and Shlomo Dubnov, Music SketchNet: Controllable Music Generation via Factorized Representations of Pitch and Rhythm, in Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020. The thesis author was the first author of this paper.

Chapter 4

Dataset

In this chapter, we introduce four datasets: Nottingham Music Database [1], Hymnal Dataset [2], Wikifornia Dataset [3], and Irish and Scottish Folk Song Dataset [37]. The links of four datasets are shown in related references. Except the Wikifornia Dataset (which might not be available now), three left datasets are available in the internet.

The Nottingham Music Database is a music collection that contains about 1000 folk tunes. It is stored in ABC notation format but with the converted MIDI files in a fixed tempo. It was first maintained by Eric Foxley, then by James Allwright. Now, the Nottingham Music Database is used in many automatic music generation or related papers [9, 10, 42]. It has fourteen sub-collections, varied from *Jigs* to *Christmas arols and Songs*. In this thesis, we use this dataset in MIDI format. Each tune file contains a melody track and a harmonic track, and we only use the melody track in the SketchVAE training.

The Hymnal Dataset is a music dataset from the website *hymnal.net*. Currently it contains about 1500 Christian hymns and spiritual songs. Since it provides users with the MIDI files of all songs, we crawl these files from the website. The Hymnal Dataset contains three sub-collections: Children, Classic, and New Songs (uncategorized). Each MIDI file contains a main melody track and a whole polyphonic track. In this thesis, we only use the main melody track in the SketchVAE training.

The Wikifornia Dataset is a relative large music collection that contains about 5000

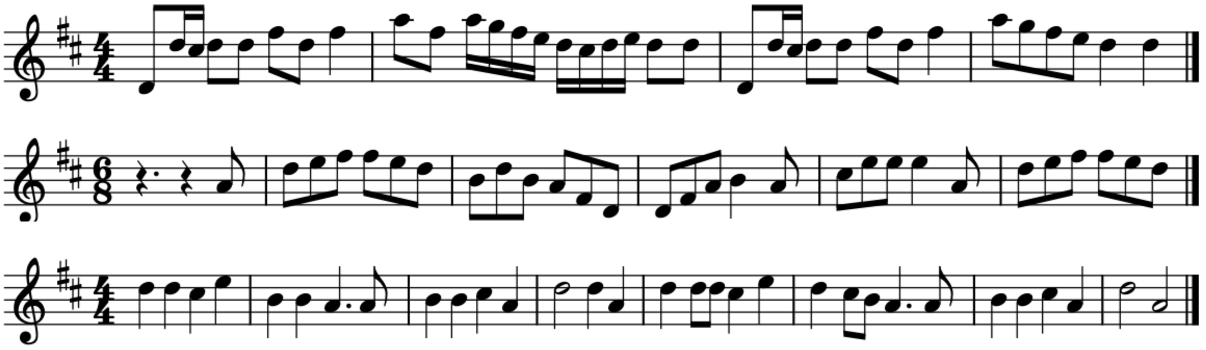

Figure 4.1. Three examples of Nottingham Music Database.

music leadsheets. It has various genres from folk, new age to pop songs. The original link of the Wikifornia Dataset is currently unavailable. But in many music library resources like MuseScore[1] and music21[2], you can still download it. Each MIDI file contains a melody track and a harmonic track, extraced from its MusicXML file. In this thesis, we only use the melody track in the SketchVAE training.

The Irish and Scottish Folk Song Dataset is a largest music collect of fours with about 20000 folk tunes. It is stored in MIDI format with only one track as the main melody. In this thesis, we use this dataset for the whole Music SketchNet training, and we base on this dataset to generate musical pieces.

Below Figures 4.1, 4.2, 4.3, 4.4 show three examples of each mentioned dataset. To make the correct alignment, we pick the similar number of measures of each example. Even though there is an end bar in each example, it does not mark the end of the music.

This chapter, in part, are currently being prepared for submission for publication of the material. Ke Chen; Taylor Berg-Kirkpatrick; Shlomo Dubnov. The thesis author will be the first author of this paper.

[1] https://musescore.org

[2] http://web.mit.edu/music21/

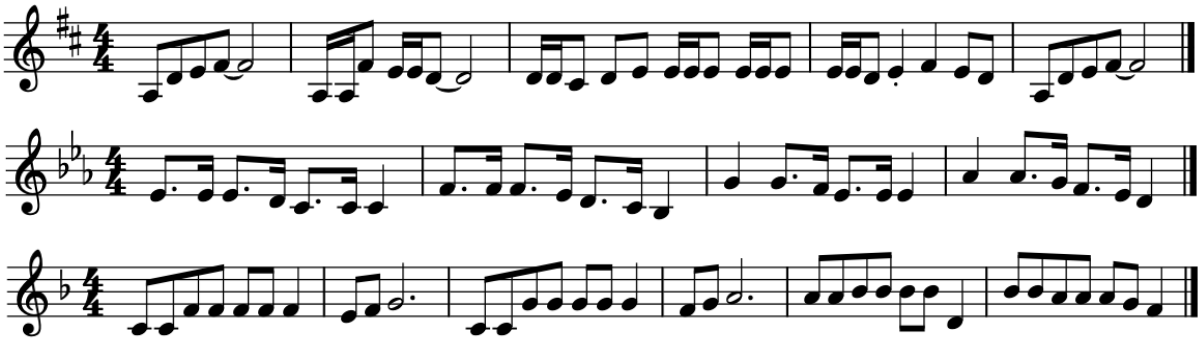

Figure 4.2. Three examples of Hymnal Music Dataset.

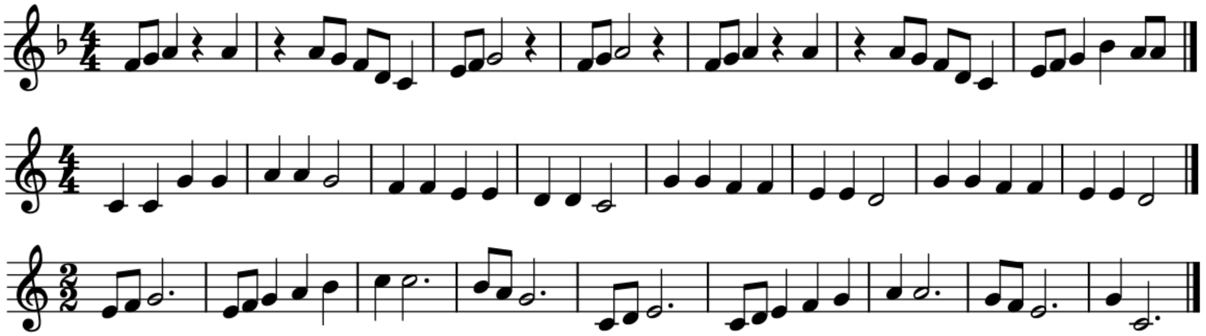

Figure 4.3. Three examples of Wikifornia Dataset.

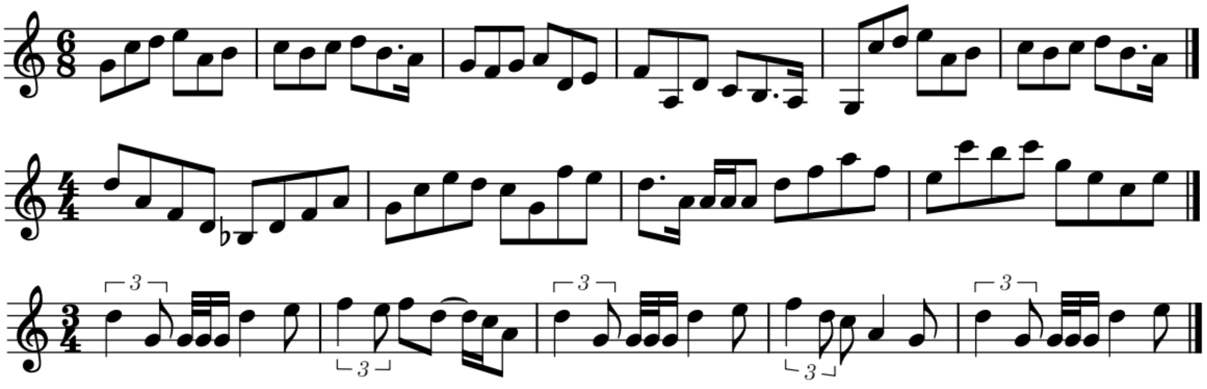

Figure 4.4. Three examples of Irish and Scottish Folk Song Dataset.

Chapter 5

SketchVAE Performance Experiment

In this chapter, we use three datasets to train and evaluate the performance of SketchVAE. As the entrance of the Music SketchNet, SketchVAE should be proven that it can accurately represent most music datasets in latent space. It contains two evaluations: (1) the reconstruction accuracy that determines if the latent vector contains enough information of the original musical measures, and (2) the approximate data likelihood that determines if the distribution of latent space accurately reflects the original distribution of the input data.

5.1 Dataset and Baseline

As described in Chapter 4, we use three datasets: the Nottingham Music Database, the Hymnal Dataset, and the Wikifornia Dataset to train and evaluate SketchVAE. they contain different genres from folk songs to classical pieces, thus we can evaluate SketchVAE's stability.

For baselines of VAE-based musical representation models, we compare SketchVAE with two existing baselines: MeasureVAE [32, 31] and EC^2-VAE [41].

MusicVAE [32] is one of the first works to apply a variational auto-encoder [29] to music. It uses a GRU-based encoder and a hierarchical decoder. The first layer of the decoder is designed to process the latent variable back to measures. And the second layer of the decoder is designed to reconstruct the notes from the upstream measures. A series of music measures can be compressed into one latent vector z for reconstruction and interpolation.

MeasureVAE [31] adopts MusicVAE's framework, but focuses on representing isolated measures and utilizes a hierarchical decoder to handle ticks and beats. Therefore, in this thesis, we take MeasureVAE as one of our comparisons (i.e. MusicVAE and MeasureVAE are in the same structure under our comparative context).

EC^2-VAE [41], as discussed in Chapter 2, factorizes music measures with separate vectors representing pitch and rhythm by using a universal encoder, a separate rhythm decoder, and a universal decoder. The latent variables can then be divided in half between pitch contour and rhythm. EC^2-VAE can perform the same function – factorizing the pitch and rhythm, as SketchVAE can perform. The difference between both is that EC^2-VAE contains one encoder and two decoders, while SketchVAE contains two encoders and one decoder.

5.2 Evaluation Metric

There are two metrics for evaluating VAE-based musical representation models: the reconstruction accuracy and the approximate data log-likelihood.

During the training, the target of SketchVAE/MeasureVAE/EC^2-VAE is to maximize the log-likelihood of the training data $\log p(x)$. As demonstrated in Chapter 2, the original $\log p(x)$ is intractable to compute for the latent representation model because it is impossible to marginalize out the latent space to obtain $\log p(x) = \sum_z \log p(x,z)$ [8, 27]. Recalling the equation in Chapter 2, we devise the Evidence Lower Bound (ELBO) as the approximate log-likelihood of the training data to make the convergence of VAE model:

$$ELBO = E[\log p_\theta(x|z)] - KL(q_\lambda(z|x)||p(z)) \leq \log p(x) \tag{5.1}$$

Even though ELBO is a tight bound for approximating the data log-likelihood, [8, 27] proposed a new method to make a tighter bound than ELBO via importance sampling. The key approach is to sample K times in the latent distribution of VAE and take the average of them as the data log-likelihood:

$$\hat{p}_K(x) = \frac{1}{K}\sum_{k=1}^{K}\frac{p(x,z_k)}{q(z_k)} = \frac{1}{K}\sum_{k=1}^{K}\frac{p_\theta(x|z_k)p(z_k)}{q_\lambda(z_k)} \tag{5.2}$$

Where $p_\theta(x|z_k)$ is the output of VAE's decoder as probability value, $p(z_k)$ is the probability of the sampled z_k in standard normal distribution, $q_\lambda(z_k)$ is the latent distribution constructed by VAE's encoder, and $z_1, z_2, ..., z_k \sim q_\lambda(z_k)$. It has been proved that $\hat{p}_K(x)$ is a tighter bound than ELBO to approximate the data log-likelihood:

$$ELBO \leq \log \hat{p}_K(x) \leq \log p(x) \tag{5.3}$$

In this thesis, we do show the detail mathematical proofs, we just use the **negative** importance sampling-based log-likelihood $-\log \hat{p}_K(x)$ as our evaluation metric to determine if the distribution of latent space accurately reflects the original distribution of the input data.

5.3 Experiment

In this section, we introduce how we conduct the experiment of SketchVAE and compare it with two baselines in detail.

For each dataset, we randomly split it into train/validate/test sets by the ratio 60%-20%-20%. Among three models (SketchVAE, MeasureVAE, and EC^2-VAE), we fix the random seed to make sure the split results are the same. As the result, 620/207/207 are selected as the train/validation/test sets in the Nottingham dataset; 1034/345/344 songs are selected as the train/validation/test sets in the Hymnal dataset; and 2680/894/894 are selected as the train/validation/test sets in the Wikifornia dataset.

For hyper-parameters of all three models, the dimension of latent variable $|z|$ is set to 256, half for the pitch contour, and the other half for the rhythm. We set the learning rate to 1e-4 and use Adam Optimization with $\beta_1 = 0.9$ and $\beta_2 = 0.998$. To make the model converge

Table 5.1. The comparison among three VAE-based musical representation models. -ELBO is the negative evidence lower bound as the loss function of VAE. -IS is the negative importance sampling-based approximation of data log-likelihood with 1 and 10 sample times.

Nottingham	**Reconstruction Acc.**	**-ELBO**	**-IS(k=1)**	**-IS(k=10)**
MeasureVAE BA	99.52%	115.82	60.43	59.90
MeasureVAE BE	98.54%	21.79	13.12	12.98
EC2VAE BA	98.90%	112.31	74.86	74.19
EC2VAE BE	97.71%	22.02	12.30	12.21
SketchVAE BA	**99.55%**	85.00	28.36	28.36
SketchVAE BE	99.46%	37.10	**8.96**	**8.50**
Hymnal	**Reconstruction Acc.**	**-ELBO**	**-IS(k=1)**	**-IS(k=10)**
MeasureVAE BA	99.65%	41.64	16.78	16.26
MeasureVAE BE	99.25%	19.11	9.32	9.20
EC2VAE BA	99.15%	42.01	17.46	17.39
EC2VAE BE	98.75%	18.75	9.01	8.96
SketchVAE BA	**99.78%**	81.79	28.34	27.40
SketchVAE BE	99.70%	32.18	**8.23**	**8.16**
Wikifornia	**Reconstruction Acc.**	**-ELBO**	**-IS(k=1)**	**-IS(k=10)**
MeasureVAE BA	99.76%	64.86	11.27	11.09
MeasureVAE BE	99.28%	19.39	**6.06**	**6.04**
EC2VAE BA	99.40%	42.70	11.05	11.00
EC2VAE BE	98.97%	20.65	7.04	7.16
SketchVAE BA	**99.80%**	66.55	25.27	24.65
SketchVAE BE	99.71%	22.99	7.94	6.64

better, we refer [24] to scale between the cross-entropy term and the KL divergence term in the loss function (i.e. ELBO). Since different models might use different scale factor, we pick each model under the best setting. The maximum number of the training epochs is 200, where all models are converged and we save the best model in the validation set and test it in the test set. Each model contains two best saved models: (1) Best Accuracy (BA) by picking the highest accuracy in the validation set, and (2) Best ELBO (BE) by picking the lowest -ELBO in the validation set.

Table 5.1 shows the results of all three models. In the evaluation columns, -ELBO refers to the negative evidence lower bound as the loss function of variational autoencoder. -IS refers to the negative importance sampling-based approximation data log-likelihood with 1 and 10 sample times. We can see -IS values are all lower than the respective -ELBO value. This proves that the importance sampling-based approximation is a tighter bound of the original data log-likelihood. All models in all three datasets can perform a confidence reconstruction accuracy around 99%, where the SketchVAE slightly beats other two models. In the evaluation of the latent space, SketchVAE has best -IS values in Nottingham and Hymnal datasets, and has a second better -IS value in the Wikifornia dataset.

From these results, we can conclude that our SketchVAE can definitely achieve the state of art in VAE-based musical representation model with around 99% reconstruction accuracy and well-formed latent distribution in various music datasets. As the first component of the Music SketchNet, the SketchVAE can store enough information of the original input music data and pass it through next components for generation.

5.4 Exchange Pitch and Rhythm

Rather than accurately reconstruct the original music measures, SketchVAE can perform a factorization of pitch and rhythm in the latent space. Since pitch and rhythm cannot be calculated and verified statistically in the latent space, we create a new music measure by exchanging the latent space elements of the two music measures, and provide three examples randomly picked from above three datasets in Figure 5.1, 5.2, 5.3. These examples can further help us verify whether pitch and rhythm are correctly factorized by SketchVAE.

Formally, given two latent music vectors z_A, z_B, representing two independent music measures. From SketchVAE, we can further get the factorized pitch vector and rhythm vector $z_A^p, z_A^r, z_B^p, z_B^r$. Therefore, we can exchange their pitch and rhythm to construct two new vectors $C = (z_A^r, z_B^p)$ and $D = (z_B^r, z_A^p)$, where one measure inherits A's rhythm pattern but with B's pitch contour, and another measure inherits reversely. And we decode C and D back to the music measure as our produced examples. This operation can be regarded as a rhythm or pitch ”analogy” from two existing music measures to new music measures.

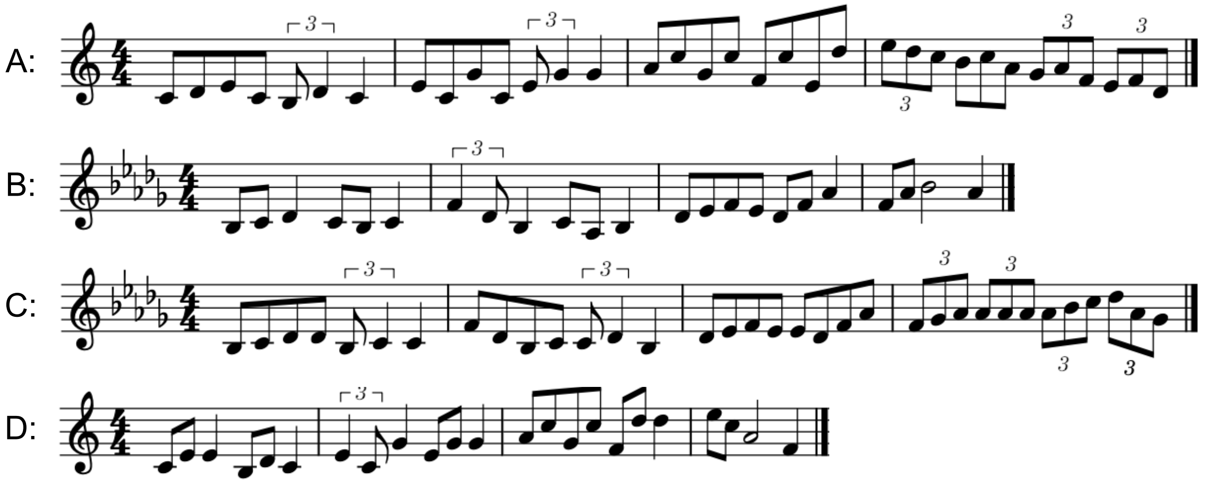

Figure 5.1. The exchange example of SketchVAE. A,B are the given original measures. C is the created measure by choosing the A's latent rhythm vector and B's latent pitch vector. C is the created measure by choosing the B's latent rhythm vector and A's latent pitch vector. Notice that the end bar does not mean the end of the music, it is set to maintain the alignment.

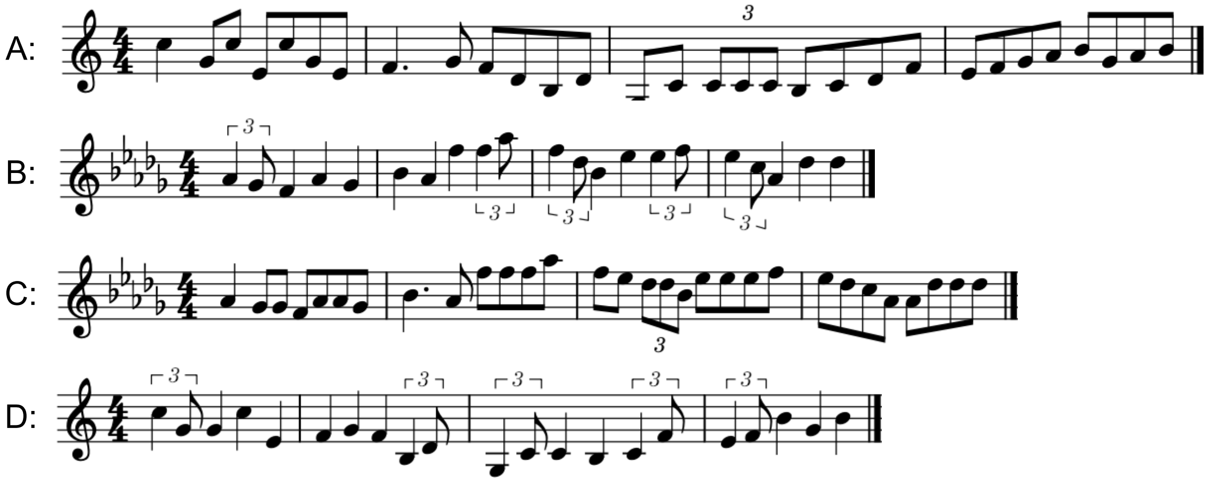

Figure 5.2. The exchange example of SketchVAE. A,B are the given original measures. C is the created measure by choosing the A's latent rhythm vector and B's latent pitch vector. C is the created measure by choosing the B's latent rhythm vector and A's latent pitch vector. Notice that the end bar does not mean the end of the music, it is set to maintain the alignment.

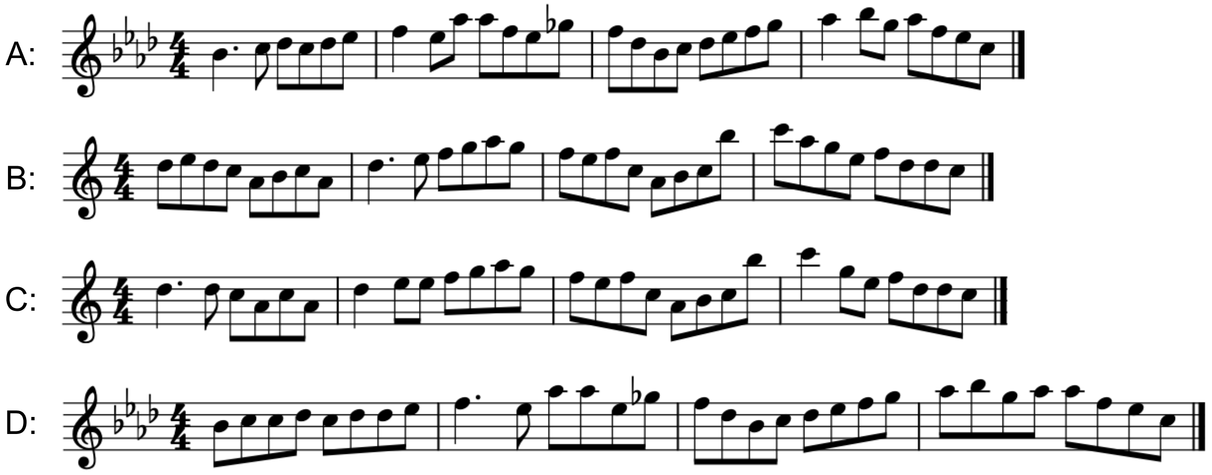

Figure 5.3. The exchange example of SketchVAE. A,B are the given original measures. C is the created measure by choosing the A's latent rhythm vector and B's latent pitch vector. C is the created measure by choosing the B's latent rhythm vector and A's latent pitch vector. Notice that the end bar does not mean the end of the music, it is set to maintain the alignment.

From three examples, all A and B measures in three examples are in different tonalities (e.g. C Major, Db Major, and Ab Major) and different rhythm patterns (e.g. the position and number of triplets). In each C measure, we can see it follows the tonality and development of B measure while having the similar rhythm pattern of A measure. In each D measure, the result is the opposite. More examples can be constructed by running the code in the link of Music SketchNet. As a conclusion, we can clearly see that the pitch and rhythm patterns are separated in the final creation of C and D measures. Both accurate reconstruction and separate factorization promised by SketchVAE proves that it is a solid structure to handle the input of the music measures and pass the information through the left components of Music SketchNet.

This chapter, in part, are currently being prepared for submission for publication of the material. Ke Chen; Taylor Berg-Kirkpatrick; Shlomo Dubnov. The thesis author will be the first author of this paper.

Chapter 6

SketchNet Generation Study

In Chapter 5, we conduct a comprehensive experiment on SketchVAE. It shows that it is a reliable representation model and can be further used in the generation scenario. In this chapter, we will use the Irish and Scottish Folk Song Dataset to train the entire Music SketchNet. As described in Chapter 4, the reason we choose this dataset is because it contains more consistent data in genres (folk tunes), and it has the largest amount of data. Due to time limitations, we believe that the full network training on this dataset is the most effective.

6.1 Baseline

Similar to the Chapter 5, we first train the SketchVAE with the Irish and Scottish Folk Song Dataset and compare it with MeasureVAE and EC^2-VAE. For SketchNet, we compare our generation results with Music InpaintNet [31], which has shown better results than the earlier baseline [22]. Similar to [31]. Further details of these models are shown in Chapter 2. In the Irish and Scottish Dataset, we select the melodies with a 4/4 time signature. About 16000 melodies are used for training and 2000 melodies each for validating and testing.

6.2 SketchVAE Measurements

6.2.1 Reconstruction

For SketchVAE, MeasureVAE and EC^2-VAE, the dimension of latent variable $|z|$ is set to 256, half for the pitch contour, and the other half for the rhythm. We set the learning rate to 1e-4 and use Adam Optimization with $\beta_1 = 0.9$ and $\beta_2 = 0.998$. Three models achieve the reconstruction accuracy of 98.8%, 98.7%, and 99.0% respectively. We can clearly conclude that all VAE models are capable of converting melodies to latent variables by achieving the accuracy around 99%. SketchVAE is capable of encoding/decoding musical materials in SketchNet.

6.3 Generation Performance

6.3.1 Training Results

The SketchNet's training is separated into stage I and II. In stage I, after training the SketchVAE, we freeze its parameters and train the SketchInpainter as shown in the right block of Figure 3.4. In stage II, with the trained SketchVAE and SketchInpainter, we freeze both, concatenate S^m with the past/future latent variables, and feed them to the SketchConnector for training.

We compare four models by using 6 measures of past and future contexts to predict 4 measures in the middle (i.e. $n_p = n_f = 6$, and $n_m = 4$). Music InpaintNet [31] is used as the baseline, along with several variations. Early stopping is used for all systems.

We compute three metrics: loss, pitch accuracy, and rhythm accuracy to evaluate the model's performance. The pitch accuracy is calculated by comparing only the pitch tokens between each generation and the ground truth (whether the model generates the correct pitch in the correct position). And the rhythm accuracy is calculated by comparing the duration and onset (regardless of what pitches it generates). The overall accuracy and loss are negatively correlated.

For this part of the experiment, we also use two special test subsets. We compute the

Table 6.1. The generation performance of different models in Irish and Scottish monophonic music dataset. The InpaintRNN is the generative network in Music InpaintNet.

	Irish-Test			Irish-Test-R			Irish-Test-NR		
Model	loss	pAcc	rAcc	loss	pAcc	rAcc	loss	pAcc	rAcc
MI.	0.662	0.511	0.972	0.312	0.636	0.975	0.997	0.354	0.959
SI.	0.714	0.510	0.975	0.473	0.619	0.981	1.075	0.374	0.964
SVSI.	0.693	0.552	0.985	0.295	0.692	0.991	1.002	0.389	0.977
SN.	**0.516**	**0.651**	**0.985**	**0.206**	**0.799**	**0.991**	**0.783**	**0.461**	**0.977**

similarities between the past and future contexts of each song in the Irish test set, pick the top 10% similar pairs (past and future contexts are almost the same) and bottom 10% pairs (almost different), and create the Irish-Test-R (repetition) and Irish-Test-NR (non-repetition) subsets.

From Table 6.1, four models are listed as comparisons: (1) MI: Music InpaintingNet [31], (2) SI: SketchVAE + InpaintNet ablation model, (3) SVSI: SketchVAE + SketchInpainter without SketchConnector, and (4) SN: the whole Music SketchNet. We can see that SketchNet beats all other models for all test sets. The performance improved more for pitch then for rhythm. The accuracy is almost the same between the 1st and 2nd model. Accuracy is slightly better if we use SketchInpainter to treat rhythm and pitch independently during generation. Lastly, with the power of transformer encoder and random unmasking process done in SketchConnector, we can achieve the best performance by using SketchNet (bottom row in Table 6.1). We further follow [6] to use the Bootstrap significance test to verify the difference between each pair's overall accuracy for models in the whole Irish-Test set (Four models, i.e. six pairs in total). The sample time is set to 10000. After calculation, all p-values except the fist and second model pair (p-value = 0.402) are less than 0.05, which proves that SketchNet is different from the left three models.

In the repetition test subsets, the loss of Music InpaintNet is 0.312, which is lower enough to capture repetitions in the musical context and fill in the missing part by copying. In most cases, copying is the correct behaviour because the original melody has repetitive pattern structures. The loss is a measurement to evaluate if the model can learn the repetitive pattern and copy

Table 6.2. Results of the subjective listening test.

Model	Complexity↑	Structure↑	Musicality↑
Original	3.22	3.47	3.56
InpaintNet	2.98	3.01	3.09
SketchNet	3.04	3.29	3.26

mechanism from the data. the SketchNet slightly outperforms InpaintNet.

6.3.2 Subjective Listening Test

However, the more interesting result is the generation with non-repetition subset. In this case, models cannot merely copy because original melodies do not repeat its content. We see higher losses in all models in this subset compared to the repetition subset. Intuitively, it means that repetitive patterns are essential to the reconstruction task, not necessarily the expressivity of the generated output would be less.

To further evaluate the proposed SketchNet, we conduct an online subjective listening test to let subjects judge the generated melodies from the non-repetition subset. Each subject will listen to three 32-second piano-rendered melodies: the original, the Music InpaintNet's generation, and the SketchNet's generation. Songs are randomly picked from the Irish-Test-NR set. The beginning and ending (past & future) are the same for the three melodies. Since the subjective feeling of music is complicated to quantify, we chose three criteria:

- The number of notes (**complexity**).
- The repetitiveness between musical structures (**structure**).
- The degree of harmony of the music (**overall musicality**).

In this way, subjects with different levels of music skills can all give reasonable answers.

Before rating the songs, subjects will see three criteria descriptions as we introduced below. The rating is ranged from 1.0 to 5.0 with a 0.5 step. We collected 318 surveyed results

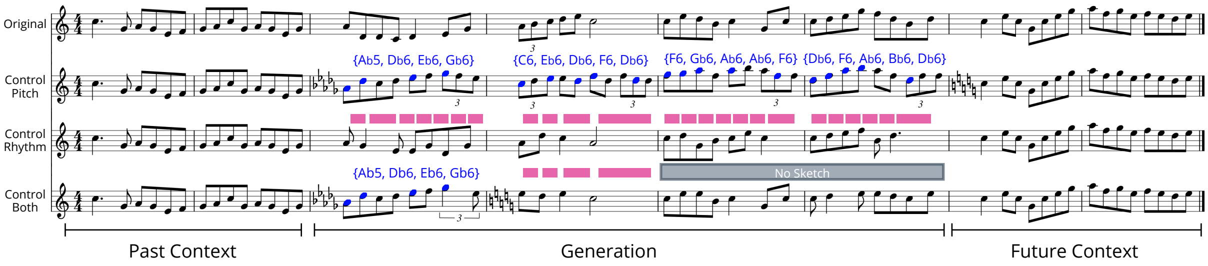

Figure 6.1. An example of sketch generation. From top to bottom: original, pitch/rhythm/mixture control. The blue pitch texts denote pitch controls, and the pink segments denote rhythm controls.

from 106 subjects (each subject listens to three groups, nine melodies in total). The average rating of each criteria for all models are shown in Table 6.2. The subjective evaluations of all three criteria in SketchNet are better for those of Music InpaintNet. Similar to section 3.3.1, we also conduct a pairwise significance test via Bootstrap in three criteria. All p-values except the ¡complexity: InpaintNet, SketchNet¿ (p-value = 0.364) are less than 0.05. It proves that three models (including original songs) are significantly different in structure and overall musicality (subjective feeling to a person). As for the complexity, we believe that the results generated by the two models are similar in terms of the richness of notes, and our model does not significantly increase the number of notes generated.

6.4 Sketch Scenario Usage

The contribution of Music SketchNet is not only shown in the performance of the generation in section 3.3, but can also be shown in the interactive scenario where users can control the generated output by specifying the rhythm or pitch contour in each measure.

Figure 6.1 shows an example of a non-repetition subset melody, where the first and last two measures are given, and the middle parts is generated. The first track is the original melody, the second track is generated with the pitch contour control, the third track is generated with the rhythm control, and the fourth track is controlled with both pitch and rhythm. We can see that each generated melody follows the control from users and develops music phrases accordingly

Table 6.3. The accuracy of the virtual control experiment.

Control Info.	Rhythm	Pitch
Pitch Acc.	0.189	**0.881**
Rhythm Acc.	**0.973**	0.848

in the missing part. Moreover, each measure is in line with the past and future measures even in the case of scale shift.

We also provide a ”virtual control experiment” to statistically show that users' control did influence the model's generation process. We randomly collect 3000 sample pairs (A, B) from the Irish-Test set. And we use the pitch/rhythm of Sample B to be the sketch information in the same missing position of Sample A. Then we let the model make the generation. We then compute the pitch/rhythm accuracy in the missing position between the generation and Song B. From 6.3 we can see if we sketch song B's rhythm into the model, the generation will follow the rhythm with 97.3% accuracy but has different (18.9%) pitches. However, when we sketch pitches, the pitches in the generation will be highly (88.1%) in line with the sketching. This proves that the user's control has a relatively high guiding effect on the result of the model generated at the specified position.

This chapter contains some materials (texts, tables, and figures) from a published conference paper: Ke Chen, Cheng-i Wang, Taylor Berg-Kirkpatrick and Shlomo Dubnov, Music SketchNet: Controllable Music Generation via Factorized Representations of Pitch and Rhythm, in Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020. The thesis author was the first author of this paper.

Chapter 7

Conclusion

In this thesis, we propose a new framework to explore decoupling latent variables in music generation. We further convert this decoupling into controllable parameters that can be specified by the user. The proposed Music SketchNet achieves the best results in the objective and subjective evaluations. Practically, we show the framework's application for the music sketching scenario where users can control the pitch contour and/or rhythm of the generated results. With this link[1], users can train, evaluate and use the Music SketchNet on any monophonic music dataset.

There are several possible extensions for this work. Music elements other than pitch and rhythm can be applied into the music sketching scenario by the latent variable decoupling. Also, how to represent a polyphonic music piece in the latent space is another pressing issue. Both are future works that can generalize this model to more applied scenarios.

Due to the number of pages and size limitation, we provide more and longer Music SketchNet samples with MIDI and audio formats in the below link[2]. And everyone could implement this structure to train with their own datasets by the code in the Github Repo.

In the future, we look forward to making a online website interface of Music SketchNet, where everyone could sketch and control the music by giving their rhythm and pitch specifications on the existing previous and future musical contexts.

[1] https://github.com/RetroCirce/Music-SketchNet.

[2] https://drive.google.com/drive/folders/1CI__Tts_YUyHCjnunqyIHVrA-IasCgUq?usp=sharing

Bibliography

[1] James Allwright. Abc version of the nottingham music database. http://abc.sourceforge.net/NMD, 2003.

[2] Anonymous. Hymal dataset. https://www.hymnal.net, 2021.

[3] Anonymous. Wikifornia dataset. http://www.wikifonia.org, 2021.

[4] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009.

[5] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pages 1171–1179, Montreal, Quebec, Canada, 2015.

[6] Taylor Berg-Kirkpatrick, David Burkett, and Dan Klein. An empirical investigation of statistical significance in NLP. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL*, pages 995–1005. ACL, 2012.

[7] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. *Deep Learning Techniques for Music Generation*. Springer, 2020.

[8] Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *4th International Conference on Learning Representations, ICLR*, 2016.

[9] Ke Chen, Gus Xia, and Shlomo Dubnov. Continuous melody generation via disentangled short-term representations and structural conditions. In *IEEE 14th International Conference on Semantic Computing, ICSC*, pages 128–135, San Diego, CA, USA, 2020. IEEE.

[10] Ke Chen, Weilin Zhang, Shlomo Dubnov, and Gus Xia. The effect of explicit structure encoding of deep neural networks for symbolic music generation. *CoRR*, abs/1811.08380, 2018.

[11] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014*

Conference on Empirical Methods in Natural Language Processing, EMNLP, pages 1724–1734, Doha, Qatar, 2014. ACL.

[12] Wikipedia contributors. Recurrent neural network. https://en.wikipedia.org/wiki/Recurrent_neural_network, 2021.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 4171–4186, Minneapolis, MN, USA, 2019. Association for Computational Linguistics.

[14] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[15] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 34–41. AAAI Press, 2018.

[16] Jeffrey L. Elman. Finding structure in time. *COGNITIVE SCIENCE*, 14(2):179–211, 1990.

[17] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 2414–2423. IEEE Computer Society, 2016.

[18] Benjamin Genchel, Ashis Pati, and Alexander Lerch. Explicitly conditioned melody generation: A case study with interdependent rnns. In *Proceedings of the 7th International Workshop on Musical Meta-creation, MUME*, 2019.

[19] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems, NIPS*, pages 2672–2680, 2014.

[20] Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*, pages 4601–4609, Barcelona, Spain, 2016.

[21] Yagmur Güçlütürk, Umut Güçlü, Rob van Lier, and Marcel A. J. van Gerven. Convolutional sketch inversion. In *Computer Vision ECCV Workshops*, pages 810–824, Amsterdam, The Netherlands, 2016. Springer.

[22] Gaëtan Hadjeres and Frank Nielsen. Anticipation-rnn: enforcing unary constraints in sequence generation, with application to interactive music generation. *Neural Computing and Applications*, 2018.

[23] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. Deepbach: a steerable model for bach chorales generation. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, pages 1362–1371, 2017.

[24] Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR*. OpenReview.net, 2017.

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

[26] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer: Generating music with long-term structure. In *7th International Conference on Learning Representations, ICLR*, New Orleans, LA, USA.

[27] Robert L. Logan IV, Matt Gardner, and Sameer Singh. On importance sampling-based evaluation of latent language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 2171–2176. Association for Computational Linguistics, 2020.

[28] Michael I. Jordan. Serial order: A parallel, distributed processing approach. In *Advances in Connectionist Theory: Speech*. Erlbaum, 1989.

[29] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR*, Banff, AB, Canada, 2014.

[30] Gareth Loy. *Composing with Computers - a Survey of Some Compositional Formalisms and Music Programming Languages*. MIT Press, 1990.

[31] Ashis Pati, Alexander Lerch, and Gaëtan Hadjeres. Learning to traverse latent spaces for musical score inpainting. In *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR*, pages 343–351, Delft, The Netherlands, 2019.

[32] Adam Roberts, Jesse H. Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. In *Proceedings of the 35th International Conference on Machine Learning, ICML*, pages 4361–4370, Stockholm, Sweden, 2018. PMLR.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, chapter 8, pages 318–362. MIT Press, 1986.

[34] Jason Sakellariou, , Francesca Tria, Loreto Vittorio, and Francois Pachet. Maximum entropy model for melodic patterns. In *ICML Workshop on Constructive Machine Learning*, 2015.

[35] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6836–6845, Honolulu, HI, USA, 2017. IEEE Computer Society.

[36] Akshay Srivatsan, Jonathan T. Barron, Dan Klein, and Taylor Berg-Kirkpatrick. A deep factorization of style and structure in fonts. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP*, pages 2195–2205. Association for Computational Linguistics, 2019.

[37] Bob L. Sturm, João Felipe Santos, Oded Ben-Tal, and Iryna Korshunova. Music transcription modelling and composition using deep learning. In *Conference on Computer Simulation of Musical Creativity, CSMC*, 2016.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems, NIPS*, pages 5998–6008, 2017.

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

[40] Kun Xu, Kang Chen, Hongbo Fu, Wei-Lun Sun, and Shi-Min Hu. Sketch2scene: sketch-based co-retrieval and co-placement of 3d models. *ACM Trans. Graph.*, 2013.

[41] Ruihan Yang, Dingsu Wang, Ziyu Wang, Tianyao Chen, Junyan Jiang, and Gus Xia. Deep music analogy via latent representation disentanglement. In *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR*, pages 596–603, Delft, The Netherlands, 2019.

[42] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 2852–2858. AAAI Press, 2017.

[43] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M. Hospedales, and Chen Change Loy. Sketch me that shoe. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 799–807, Las Vegas, NV, USA, 2016. IEEE Computer Society.