LEARNING INTERPRETABLE REPRESENTATION FOR CONTROLLABLE POLYPHONIC MUSIC GENERATION

First Author
Affiliation1
author1@ismir.edu

Second Author
Retain these fake authors in submission to preserve the formatting

Third Author
Affiliation3
author3@ismir.edu

ABSTRACT

While deep generative models have become the leading methods for algorithmic composition, it remains a challenging problem to *control* the generation process because the hidden variables of most deep-learning models lack good interpretability. Inspired by the content-style disentanglement idea, we design a novel architecture, under the VAE framework, that effectively learns two interpretable latent factors of polyphonic music: chord and texture. The current model focuses on learning 8-beat long piano composition segments. We show that such chord-texture disentanglement provides a controllable generation pathway leading to a wide spectrum of applications, including compositional style transfer, texture variation, and accompaniment arrangement. Both objective and subjective evaluations show that our method achieves a successful disentanglement and high quality controlled music generation. All demos can be accessed via the demo folder.

1 Introduction

With the development of artificial neural networks, deep learning has become the leading technique for automated music generation. In specific, we see recurrent and attention-based models being able to generate more musical and human-like music compared to traditional time-series models and rule-based algorithms. However, a main drawback of these deep generative models is that they behave like "black boxes", and it is difficult to interpret the musical meaning of their internal latent variables. Consequently, it remains a challenging task to control the generation process (i.e., to guide the music flow by manipulating the compositional factors such as chord progression, melody, and texture style). This limitation restricts the application scenario of the powerful models.

In this paper, we improve the model interpretability for music generation via constrained representation learning. Inspired by the content-style disentanglement idea [8], we enforce the model to learn two fundamental factors of polyphonic music: *chord* (content) and *texture* (style). The current design focuses on learning 8-beat long piano composition segments under a variational autoencoder (VAE) framework.

The core of the model design lies in the encoder. We incorporated the encoder with two **inductive biases** for a successful **chord-texture disentanglement**. The former applies a rule-based chord recognizer and embeds the information into the first half of the hidden representation. The latter regards music as 2-D images and uses a tailored convolutional network to extract the texture information, storing it into the second half of the hidden representation. As for the decoder, we adopted the design from PianoTree VAE [1], an architecture that can reconstruct polyphonic music from the hidden representation in a hierarchical manner.

We further show that the interpretable representations are **general-purpose**, empowering a wide spectrum of controllable music generation. In this study, we explore the following three scenarios:

Task 1: Compositional style transfer by swapping the chord and texture factors of different pieces of music, which can help us re-harmonize or re-arrange a piece of music following the style of another piece.

Task 2: Texture variation by sampling the texture while keeping the chords, which is analogous to the creation of "Theme and Variations" form of composition.

Task 3: Accompaniment arrangement by predicting the texture given the melody using a downstream seq2seq generative model. This task is similar to the creation of "cover songs".

In sum, the contributions of our paper are as follows:

- We designed the first representation disentanglement method for polyphonic music, which learns two interpretable factors: chord and texture.
- We show that the interpretable factors are general-purpose features for controllable music generation, which reduces the necessity to design heavily-engineered control-specific model architectures. As far as we know, this is the first attempt to explicitly control the texture feature for polyphonic music generation.
- We demonstrate that control methods are effective and the quality of generated music is high. Some style transferred pieces are rated even higher than the original ones composed by humans.

2 Related Work

We review two techniques of automated music generation related to our paper: controlled generation (in Section 2.1) and representation disentanglement (in Section 2.2). For a more general review of deep music generation, we refer readers to [3, 4].

2.1 Controlled Music Generation

Most existing learning-based methods regard controlled music generation a *conditional estimation* problem. That is, to model $p(\text{music}|\text{control})$, in which both music and control are usually time-series features. Another approach that is closely related to conditional estimation is to first learn the joint distribution $p(\text{music}, \text{control})$ and later on *force* the value of control during the generation process.

The above two methods have been used in various tasks, including generating chords based on the melody [31], creating the melody based on the chords [6, 36], completing the counterparts or accompaniment based on the melody or chord [10–12, 15, 32, 38], and producing the audio waveform based on timbre features [14, 20].

However, there are two serious limitations. First, the control is too rigid because it relies on discrete or symbolic labels in most cases. Second, many abstract music factors, such as texture and melody contour, could hardly be explicitly coded by labels. Consequently, it remains a challenging task to control music by more abstract factors without complex heuristics [22].

2.2 Music Representation Disentanglement

Learning disentangled representations is an ideal solution to the two problems above, since: 1) representation learning embeds discrete music and control sequences into a continuous latent space , and 2) disentanglement techniques can further decompose the latent space into interpretable subparts that correspond to abstract music factors. Recent studies show that VAEs [18, 30] are in general an effective framework to learn the representations of discrete music sequences, and the key to a successful disentanglement is to incorporate proper inductive biases into the representation learning models [23].

Under a VAE framework, an inductive bias can be realized in various forms, including constraining the encoder [24], constraining the decoder [7], imposing multi-task loss functions [5, 37], and enforcing transformation invariant results during the learning process [21, 25]. Among these studies, Deep Music Analogy [37] is most relevant to this paper as it can disentangle pitch and rhythm factors for monophonic segments. Our paper extends this idea to polyphonic composition but the model design is very different.

3 Model

In this section, we introduce the model design and data representation in detail. The goal is to learn the representations of 8-beat long piano compositions (with $\frac{1}{4}$ beat as shortest unit) and disentangle the representations into two interpretable factors: chord and texture.

Figure 1 shows the overall architecture of the model.

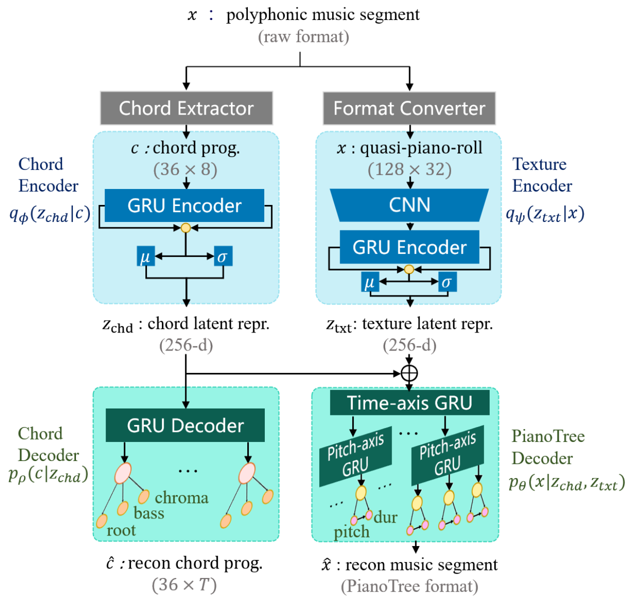


Figure 1: The model diagram.

It adopts a VAE framework and contains four parts: 1) a chord encoder, 2) a chord decoder, 3) a texture encoder, and 4) a PianoTree decoder. The chord encoder and chord decoder can be seen as a standalone VAE which extracts the latent chord representation z_{chd}. On the other hand, the texture encoder aims to extract the texture representation z_{txt} using a chord-invariant convolutional mapping. Finally, the PianoTree decoder takes in both z_{chd} and z_{txt} and outputs the original music in a tree-structure data format.

3.1 Chord Encoder

The chord encoder first applies rule-based methods [27, 29] to extract the chord progression under one-beat resolution. Each extracted chord progression is a 36 by 8 matrix, where each column denotes a chord of one beat. Each chord is a 36-D vector consisting of three parts: a 12-D one-hot vector for the pitch class of the *root*, a 12-D one-hot vector for the *bass*, and a 12-D multi-hot *chroma* vector.

The chord progression is then fed into a bi-directional GRU encoder [30], and the last hidden states on both ends of the GRU are concatenated and used to approximate the posterior distribution of z_{chd}. Following the assumption of a standard VAE, z_{chd} has a standard Gaussian prior and follows an isotropic Gaussian posterior.

Note that although the chord progression here is extracted using algorithms, it can also be provided by external labels, in which case the whole model becomes a conditional VAE [35].

3.2 Chord Decoder

The chord decoder reconstructs the chord progression from z_{chd} using another bi-directional GRU. The reconstruction loss of a chord progression is computed as a summation of 8 beat-wise chord loss using cross entropy functions [9]. For each beat, the chord loss is defined as the product of three parts: 1) the root loss, 2) the bass loss, and 3) the chroma loss. The root and bass are both considered 12-way categorical distributions and a chroma is regarded as 12 independent Bernoulli distributions.

3.3 Texture Encoder

The input of the texture encoder is an 8-beat segment of polyphonic composition represented by an image-like data format slightly modified from the piano-roll [11]. Each 8-beat segment is represented by a 128 by 32 matrix, where each row corresponds to a MIDI pitch and each column corresponds to $\frac{1}{4}$ beat. The data entry at (p, t) records the duration of the note if there is a note onset, and zero otherwise.

The texture encoder aims to learn a chord-invariant representation of texture by leveraging the translation invariance property of convolutional neural networks (CNNs) [19]. We use a convolutional layer with kernel size 12×4 and stride $(1, 4)$, which is followed by a ReLU activation [26] and max pooling with kernel size 4×1 and stride $(4, 1)$. The convolutional layer has one input channel and 10 output channels. The convolutional layer design aims at extracting a blurry "concept sketch" of the polyphonic texture which contains minimum information of the underlying chord. The idea is that when such blurry sketch are combined with specific chord representation, the decoder can identify its concrete pitches in a musical way.

The output of the convolutional layer is then fed into a bi-directional GRU encoder to extract the texture representation z_{txt}, similar to how we encode z_{chd} introduced in Section 3.1.

3.4 PianoTree Decoder

The PianoTree decoder takes the concatenation of z_{chd} and z_{txt} as input and decodes the music segment using the same decoder structure invented in PianoTree VAE [1], a hierarchical model structure for polyphonic representation learning. The decoder works as follows. First, it generates 32 frame-wise hidden states (one for each $\frac{1}{4}$ beat) using a GRU layer. Then, each frame-wise hidden state is further decoded into the embeddings of individual notes using another GRU layer. Finally, the pitch and duration for each note are reconstructed from the note embedding using a fully-connected layer and a GRU layer, respectively. For more detailed derivation and model design, we refer the readers to [1].

3.5 Training Objective

We denote x as the input music segment and $c = f(x)$ the chord progression extracted by algorithm $f(\cdot)$. We assume the priors $p(z_{\text{chd}})$ and $p(z_{\text{txt}})$ are standard Gaussian. We denote the output posterior distributions of chord encoder and texture encoder as $q_\phi(z_{\text{chd}}|c)$, $q_\psi(z_{\text{txt}}|x)$, the outputs of chord decoder and PianoTree decoder as $p_\rho(c|z_{\text{chd}})$ and $p_\theta(x|z_{\text{chd}}, z_{\text{txt}})$. The objective of the model is:

$$\begin{aligned}
&\mathcal{L}(\phi, \psi, \rho, \theta; x) = \\
&- \mathbb{E}_{\substack{z_{\text{chd}} \sim q_\phi \\ z_{\text{txt}} \sim q_\psi}} \left[\log p_\rho(c|z_{\text{chd}}) + \log p_\theta(x|z_{\text{chd}}, z_{\text{txt}})\right] \\
&+ \mathrm{KL}(q_\phi||p(z_{\text{chd}})) + \mathrm{KL}(q_\psi||p(z_{\text{txt}})). \qquad (1)
\end{aligned}$$

4 Controlled music Generation

In this section, we show some controlled generation examples of the three tasks mentioned in the introduction.

4.1 Compositional Style Transfer

By regarding chord progression *content* and texture *style*, we can achieve compositional style transfer by swapping the texture representations of different pieces. Figure 2 shows the transferred results ((c) & (d)) based on two 16-bar samples ((a) & (b)) in the test set by swapping z_{txt} every 2 bars. [1]

We see that such long-term style transfer is successful: The generated segment (c) follows the chord progression of (b) while mimicking the texture of (a), while (d) follows the chord progression of (a) while mimicking the texture of (b). As shown in the marked scores, the style transfer is very effective. For example, the cut-offs, melody contours, and the shape of the left-hand accompaniment are all preserved. The audio samples of these examples can be found at the demo folder.

4.2 Texture Variation by Sampling

We can make variations of texture by sampling from z_{txt} while keeping z_{chd}. Here, we investigate two sampling strategies: sampling from the posterior $q(z_{\text{txt}}|x)$, and sampling from the prior $p(z_{\text{txt}})$.

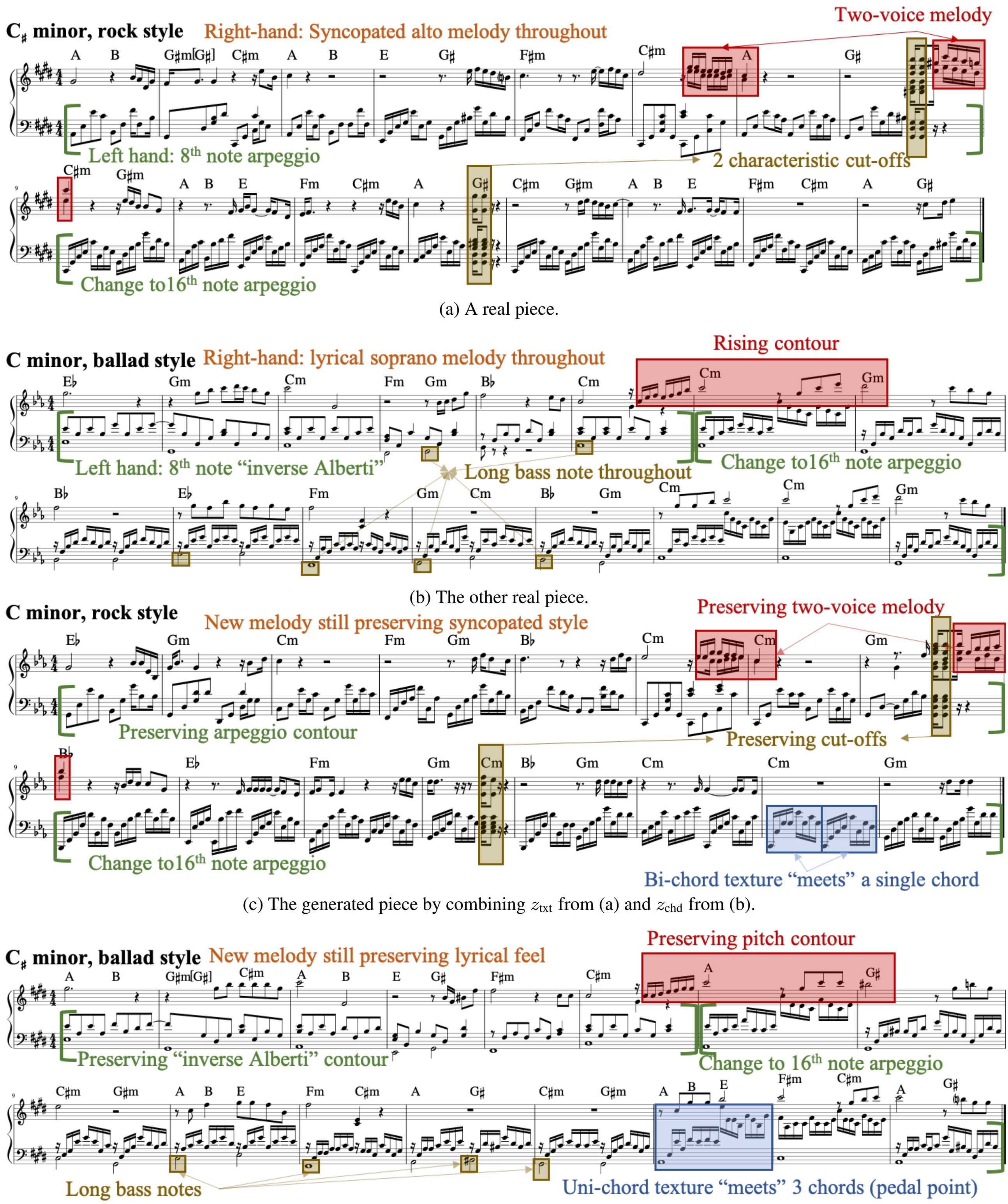
Sampling from the posterior distribution $q(z_{\text{txt}}|x)$ yields reasonable variations as shown in Figure 3a. On one hand, the variations of right-hand melody has the effect of improvisation according to the chord progression and melody contour. On the other hand, there are only tiny changes in the left hand part, showing that the model regards the left hand accompaniment as the dominant feature of texture.

Sampling from the prior distribution $p(z_{\text{txt}})$ changes the texture completely. Figure 3b shows a series of examples of prior sampling under the same chord progression C-Am-F-G. The resulting generations follow exactly the chord progression but with new textures. The audio samples of these examples can be found at the demo folder

4.3 Accompaniment Arrangement

We use a downstream predictive model to achieve accompaniment arrangement: generating 16-bar piano accompaniment *conditioned* on melody and chord progression. As suggested in [28], we first encode the data every 2 bars. For the accompaniment, we use the (pretrained) proposed model to compute the latent chord and texture representation, denoted by $\mathbf{z}_{\text{chd}} = [z_{\text{chd}}^{(1)}, ..., z_{\text{chd}}^{(4)}]$ and $\mathbf{z}_{\text{txt}} = [z_{\text{txt}}^{(1)}, ..., z_{\text{txt}}^{(4)}]$. For the melody, we use pretrained EC^2VAE [37] to compute the latent pitch and rhythm representations every 2 bars, denoted by $\mathbf{z}_{\text{p}} = [z_{\text{p}}^{(1)}, ..., z_{\text{p}}^{(4)}]$ and $\mathbf{z}_{\text{r}} = [z_{\text{r}}^{(1)}, ..., z_{\text{r}}^{(4)}]$. Then, we adopt a vanilla Transformer [33] to model $p(\mathbf{z}_{\text{txt}}|\mathbf{z}_{\text{p}}, \mathbf{z}_{\text{r}}, \mathbf{z}_{\text{chd}})$, in which the encoder takes in the condition and the decoder's input is a shifted right version $\mathbf{z}_{\text{txt}}$. Both encoder and decoder inputs are incorporated with a *positional encoding* indicating the time positions and a learned *factor embedding* indicating

[1] The presented score are converted from MIDI by the authors. The chord labels are inferred from the original/generated samples.



(a) A real piece.

(b) The other real piece.

(c) The generated piece by combining z_{txt} from (a) and z_{chd} from (b).

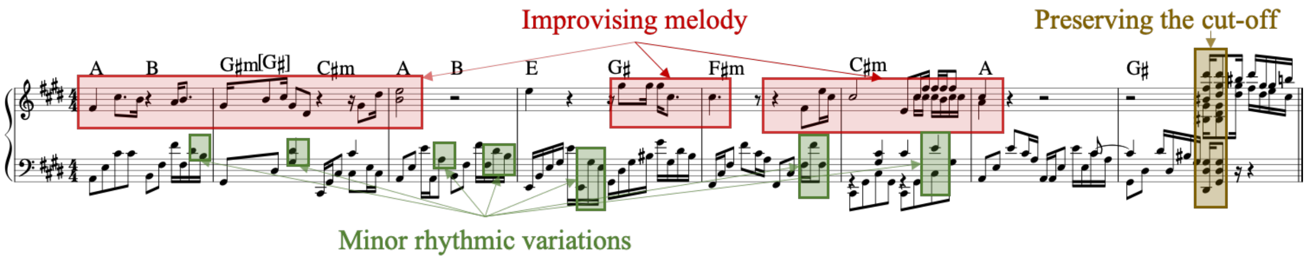
(d) The generated piece by combining z_{txt} from (b) and z_{chd} from (a).

Figure 2: An example of compositional style transfer of 16-bar-long samples when $k = 2$.

the representation type (i.e., pitch, rhythm, chord or texture).

Figure 4 shows an example of accompaniment arrangement, where the first staff shows the melody and the second staff shows the piano accompaniment. Here, the whole melody, together with the chord progression and the first 2 bars of accompaniment are given. The model predicts a similar texture to the given accompaniment and predicts a secondary melody line as a transition when the lead melody is rest.

Note that the arrangement can be generated in a flexible way by conditioning on different sets of latent fac-



(a) An example of posterior sampling of z_{txt} of the first 8 bars of the segment (a) in Figure 2

(b) An example of prior sampling of z_{txt} under given chord progression C-Am-F-G. Each two-bar segment is independently sampled, having different texture.

Figure 3: Examples of texture variations via posterior sampling and prior sampling.

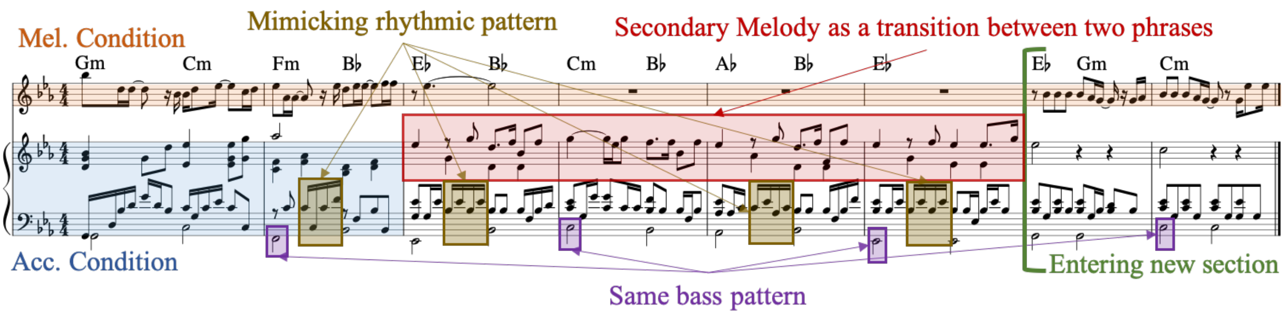


Figure 4: An example of accompaniment arrangement conditioned on melody, chord progression, and first 4 bars of accompaniment.

tors. We here only present one possibility to control the arrangement using chord, melody, and a short accompaniment context. Much longer examples and more conditioning settings are available in the demo folder.

5 Experiments

5.1 Dataset and Training

We train our model on the self-collected dataset from the internet [2], which contains about 1K MIDI files of pop songs. We further extract the chord annotations using [27, 29]. We only keep the pieces with $\frac{2}{4}$ and $\frac{4}{4}$ meters and cut them into 8-beat music segments (so that each data sample in our experiment contains 32 time steps under 16$^{\text{th}}$ note resolution). In all, we have 66K samples and we randomly select 90% for training and 10% for testing. All samples are further augmented by -6 to 5 semitones.

In our experiment, the VAE model uses 256, 512, and 512 hidden dimensions for the GRUs in chord encoder, chord decoder and texture encoder respectively. The latent dimension of z_{chd} and z_{txt} are both 256. The model size of the PianoTree decoder is the same as the implementation in the original paper [1]. The transformer model has the following size: outputs of dimension $= 256$, number of layers $= 4$ and number of heads $= 8$.

For both models, we use Adam optimizer [17] with a secheduled learning rate from 1e-3 to 1e-5. Moreover, for the VAE model, we use KL-annealing [2], i.e. setting a weight parameter for the KL-divergence loss starting from 0 to 0.1. We set batch size to be 128 and the training converges within 6 epoches. For the downstream transformer model, we use 12K warmup steps of learning rate as suggested by [34]. We use the same model size and the model converges within 40 epoches.

5.2 Objective Measurement

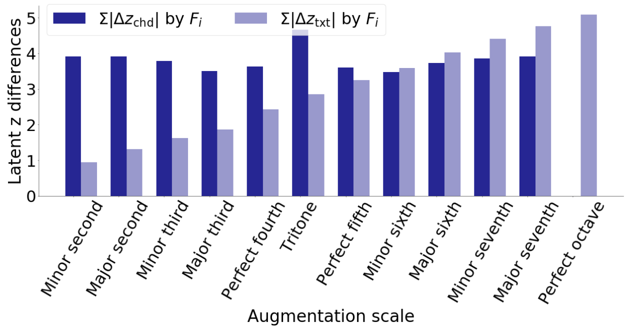
When z_{chd} andz_{txt} are well disentangled, small variations over the note pitches of the original music should lead to a larger change on z_{chd}, while variations of rhythm will influence more on z_{txt}. Following this assumption, we adopt a *disentanglement evaluation via data augmentation* method used in [16] and further developed in [37].

We define F_i as the operation of transposing all the notes by i semitones, and use the L_1-norm to measure the change of latent z after augmentation. Figure 5a shows a comparison between $\Sigma|\Delta z_{\text{chd}}|$ and $\Sigma|\Delta z_{\text{txt}}|$ when we apply F_i to all the music pieces in the test set (where $i \in [1, 12]$).

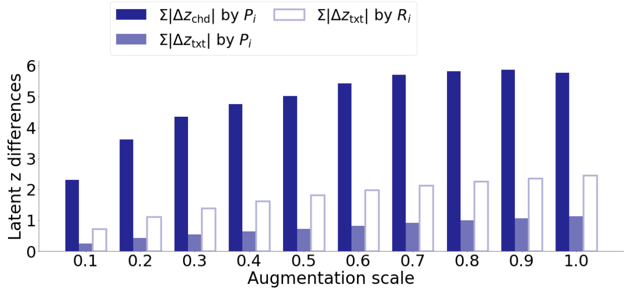
It is conspicuous that when augmenting pitch in a small range, the change of z_{chd} is much larger than the change of

[2] Anonymous dropbox link.

z_{txt}. At the same time, the change of z_{txt} gets higher as the augmentation scale increases. Similar to the result in [37], the change of z_{chd} reflects human pitch perception as z_{chd} is very sensitive to a tritone transposition, and least sensitive for a perfect octave.



(a) A comparison between Δz_{chd}, Δz_{txt} after pitch transposition on all notes.



(b) A comparison among Δz_{chd}, Δz_{txt} after beat-wise pitch transposition and texture augmentation with different probabilities.

Figure 5: Results of objective measurement.

We further define P_i as the function to randomly transpose all the notes in one beat either up or down one semitone under a certain probability i, and R_i as the function to randomly reduce the note duration by half. Figure 5b shows a comparison between $\Sigma|\Delta z_{\text{chd}}|$ and $\Sigma|\Delta z_{\text{txt}}|$ when we apply P_i and R_i to all the music pieces in our test set (where $i \in [0.1, 1.0]$).

For each value of i in the figure 5b, the first and second bars demonstrate $\Sigma|\Delta z_{\text{chd}}|$ and $\Sigma|\Delta z_{\text{txt}}|$ caused by P_i function, while the third bar indicates $\Sigma|\Delta z_{\text{txt}}|$ caused by R_i function. (We did not show $\Sigma|\Delta z_{\text{chd}}|$ caused by R_i since they are all zero.) It again proves that the chord representation is affected more by pitch variations compared to texture variations. Also, it shows that z_{txt} is more sensitive to rhythm variation than pitch variation.

5.3 Subjective Evaluation

Besides objective measurement, we conduct a survey to evaluate the musical quality of compositional style transfer (see 4.1). Each subject listen to 10 2-bar pieces with different chord progressions, each paired with 5 style-transfer versions generated by swapping the texture representation with a random sample from the test set. In other words, each subject evaluates 10 groups of samples, each of which contains 6 versions of textures (1 from original piece and 5 from other pieces) under the same chord progression. Both the order of groups and the sample order within each group are randomized. After listening to each sample, subjects rate them based on a 5-point scale from 1 (very low) to 5 (very high) according to three criteria: creativity, naturalness and musicality.

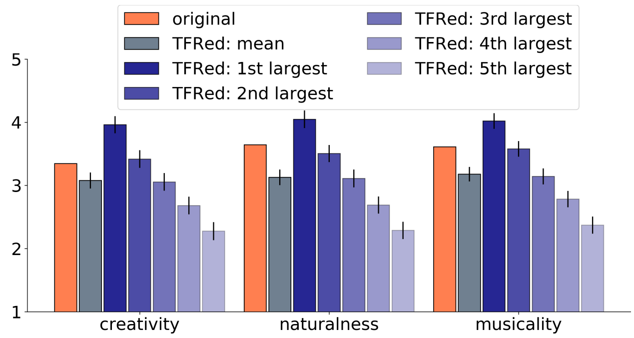


Figure 6: Subjective evaluation results. Here "TFRed: xth largest" denotes the x^{th} (largest) order statistic of the transferred segments.

A total of 36 subjects (26 female and 10 male) participated in the survey. Figure 6 shows the comparison result among the original pieces (indicated by the orange bars) and the transferred pieces in terms of their mean and *order* statistics. The heights of bars represent averaged ratings across the subjects and the error bars represent the confidence intervals computed via paired t-test [13]. The result shows *if we randomly transfer a piece's texture 5 times, the best result is significantly better than the original version (with p-value* < 0.005*)*, and there are only marginal differences between the second largest statistics and the original (with p-vale > 0.05) in terms of creativity and musicality. We also see that on average the transferred results are still rated lower than the original ones. How to automatically decide the quality of a transferred result is considered a future work.

6 Conclusion and Future Work

In conclusion, we contributed an effective algorithm to disentangle polyphonic music representation into two interpretable factors, chord and texture, under a VAE framework. Such interpretable representations serve as an intuitive human-computer co-creation interface, by which we can precisely manipulate individual factors to control the flow of the generated music. In this paper, we demonstrated three ways to interact with model, including compositional style transfer via swapping the latent codes, texture variation by sampling from the latent distribution, accompaniment arrangement using downstream conditional prediction, and there are potentially many more. We hope this work can shed light on the field of controllable algorithmic composition in general, especially on the paradox between model complexity and model interpretability.

We are acknowledged that the learned music factors are still very basic. In the future, we plan to extract more abstract and longer-range features using hierarchical models. We also plan to explore more ways to control the music generation for practical usage.

7 References

[1] A.Anonymous. Pianotree vae: Structured representation learning for polyphonic music. *a conference paper under review*, 2020. 1, 3.4, 5.1

[2] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015. 5.1

[3] Jean-Pierre Briot. From artificial neural networks to deep learning for music generation–history, concepts and trends. *arXiv preprint arXiv:2004.03586*, 2020. 2

[4] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. Deep learning techniques for music generation–a survey. *arXiv preprint arXiv:1709.01620*, 2017. 2

[5] Gino Brunner, Andres Konrad, Yuyi Wang, and Roger Wattenhofer. Midi-vae: Modeling dynamics and instrumentation of music with applications to style transfer. *arXiv preprint arXiv:1809.07600*, 2018. 2.2

[6] Ke Chen, Weilin Zhang, Shlomo Dubnov, Gus Xia, and Wei Li. The effect of explicit structure encoding of deep neural networks for symbolic music generation. In *2019 International Workshop on Multilayer Music Representation and Processing (MMRP)*, pages 77–84. IEEE, 2019. 2.1

[7] Keunwoo Choi and Kyunghyun Cho. Deep unsupervised drum transcription. *arXiv preprint arXiv:1906.03697*, 2019. 2.2

[8] Shuqi Dai, Zheng Zhang, and Gus G. Xia. Music style transfer: A position paper. *arXiv preprint arXiv:1803.06841*, 2018. 1

[9] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005. 3.2

[10] Chris Donahue, Huanru Henry Mao, Yiting Ethan Li, Garrison W Cottrell, and Julian McAuley. Lakhnes: Improving multi-instrumental music generation with cross-domain pre-training. *arXiv preprint arXiv:1907.04868*, 2019. 2.1

[11] 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 *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 2.1, 3.3

[12] 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-Volume 70*, pages 1362–1371. JMLR. org, 2017. 2.1

[13] Henry Hsu and Peter A Lachenbruch. Paired t test. *Encyclopedia of Biostatistics*, 6, 2005. 5.3

[14] Sicong Huang, Qiyang Li, Cem Anil, Xuchan Bao, Sageev Oore, and Roger B Grosse. Timbretron: A wavenet (cyclegan (cqt (audio))) pipeline for musical timbre transfer. *arXiv preprint arXiv:1811.09620*, 2018. 2.1

[15] Yu-Siang Huang and Yi-Hsuan Yang. Pop music transformer: Generating music with rhythm and harmony. *arXiv preprint arXiv:2002.00212*, 2020. 2.1

[16] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arXiv preprint arXiv:1802.05983*, 2018. 5.2

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5.1

[18] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2.2

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 3.3

[20] Ohsung Kwon, Inseon Jang, ChungHyun Ahn, and Hong-Goo Kang. Emotional speech synthesis based on style embedded tacotron2 framework. In *2019 34th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)*, pages 1–4. IEEE, 2019. 2.1

[21] Stefan Lattner, Monika Dörfler, and Andreas Arzt. Learning complex basis functions for invariant representations of audio. *arXiv preprint arXiv:1907.05982*, 2019. 2.2

[22] Stefan Lattner, Maarten Grachten, and Gerhard Widmer. Imposing higher-level structure in polyphonic music generation using convolutional restricted boltzmann machines and constraints. *arXiv preprint arXiv:1612.04742*, 2016. 2.1

[23] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. *arXiv preprint arXiv:1811.12359*, 2018. 2.2

[24] Yin-Jyun Luo, Kat Agres, and Dorien Herremans. Learning disentangled representations of timbre and pitch for musical instrument sounds using gaussian mixture variational autoencoders. *arXiv preprint arXiv:1906.08152*, 2019. 2.2

[25] Michael F Mathieu, Junbo Jake Zhao, Junbo Zhao, Aditya Ramesh, Pablo Sprechmann, and Yann LeCun.

Disentangling factors of variation in deep representation using adversarial training. In *Advances in neural information processing systems*, pages 5040–5048, 2016. 2.2

[26] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010. 3.3

[27] Bryan Pardo and William P Birmingham. Algorithms for chordal analysis. *Computer Music Journal*, 26(2):27–49, 2002. 3.1, 5.1

[28] Ashis Pati, Alexander Lerch, and Gaëtan Hadjeres. Learning to traverse latent spaces for musical score inpainting. *arXiv preprint arXiv:1907.01164*, 2019. 4.3

[29] Colin Raffel, Brian McFee, Eric J Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, Daniel PW Ellis, and C Colin Raffel. mir_eval: A transparent implementation of common mir metrics. In *In Proceedings of the 15th International Society for Music Information Retrieval Conference, ISMIR*. Citeseer, 2014. 3.1, 5.1

[30] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. *arXiv preprint arXiv:1803.05428*, 2018. 2.2, 3.1

[31] Ian Simon, Dan Morris, and Sumit Basu. Mysong: automatic accompaniment generation for vocal melodies. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 725–734, 2008. 2.1

[32] Ian Simon, Adam Roberts, Colin Raffel, Jesse Engel, Curtis Hawthorne, and Douglas Eck. Learning a latent space of multitrack measures. *arXiv preprint arXiv:1806.00195*, 2018. 2.1

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 4.3

[34] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. *arXiv preprint arXiv:2002.04745*, 2020. 5.1

[35] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, pages 776–791. Springer, 2016. 3.1

[36] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. Midinet: A convolutional generative adversarial network for symbolic-domain music generation. *arXiv preprint arXiv:1703.10847*, 2017. 2.1

[37] Ruihan Yang, Dingsu Wang, Ziyu Wang, Tianyao Chen, Junyan Jiang, and Gus Xia. Deep music analogy via latent representation disentanglement. *arXiv preprint arXiv:1906.03626*, 2019. 2.2, 4.3, 5.2

[38] Hongyuan Zhu, Qi Liu, Nicholas Jing Yuan, Chuan Qin, Jiawei Li, Kun Zhang, Guang Zhou, Furu Wei, Yuanchun Xu, and Enhong Chen. Xiaoice band: A melody and arrangement generation framework for pop music. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2837–2846, 2018. 2.1