STYLE-CONDITIONED MUSIC GENERATION

Yu-Quan Lim[⋆] *Chee Seng Chan*[⋆] *Fung Ying Loo*[†]

[⋆]Centre of Image & Signal Processing, Fac. Comp. Sci. & Info. Tech., University of Malaya, Malaysia
[†]Department of Music, University of Malaya, Malaysia
{yuquan95@gmail.com, cs.chan@um.edu.my, loofy@um.edu.my}

ABSTRACT

Recent works have shown success in generating music using a Variational Autoencoder (VAE). However, we found out that the style of the generated music is usually governed or limited by the training dataset. In this work, we proposed a new formulation to the VAE that allows users to condition on the style of the generated music. Technically, our VAE consists of two latent spaces - *content* and *style* space to encode the content and style of a song separately. Each style is represented by a continuous style embedding, unlike previous works which mostly used discrete or one-hot style labels. We trained our model on public datasets that made up of Bach chorales and western folk tunes. Empirically, as well as from music theory point of view, we show that our proposed model can generate better music samples of each style than a baseline model. The source code and generated samples are available at `https://github.com/daQuincy/DeepMusicvStyle`

Index Terms— music synthesis, deep learning, style transfer

1. INTRODUCTION

The advancement of deep neural networks (DNN), in particular Variational Autoencoder (VAE) and Generative Adversarial Networks (GAN) is blurring the line between art and science. Recent efforts have demonstrated the ability of DNN to generate various forms of creative contents with decent if not good quality, such as artistic images [1], poems [2] and music [3]. As a result of this, musicians and artists have been able to leverage on the ability of DNN to extend their ideas to produce creative contents [4,5]. These deep learning applications could be better utilized if the users could have more controls over the model. In this paper, we explore the idea of a generative model that can generate symbolic music conditioned on a compositional style, selected by the user.

In the visual domain, researchers have shown success in developing generative models that allow users to specify the style, content and even varying subtle details of the generated image [6, 7]. Sequence generation, in particular music generation, is however still slightly lagging behind the visual counterpart. Recent advances have shown some success in music generative models that allow user control [8–11]. For instance, most of these models are interactive in such a way that it allows conditioning on attributes such as chord progression, note density and rhythmic style. The compositional style of the generated music, however, is mostly governed (limited) by the type of music in the training dataset, such as Bach chorales, pop music and jazz music.

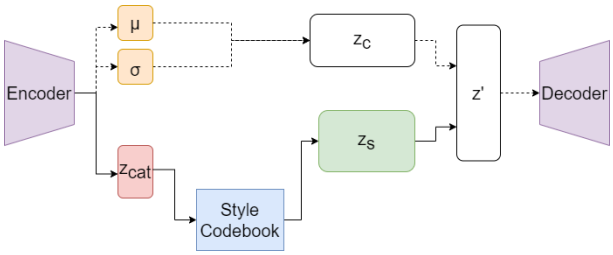

Fig. 1. Our proposed model architecture. z_c embeds the "content" of the input song while z_{cat} is a categorical variable denoting the style. The embedding for the respective style is then being retrieve from the style codebook, yielding z_s. Finally, z_c and z_s are concatenated together to form z for the decoder to synthesize a new song.

To effectively allow users to condition on the style of the synthesized music, our model innovates two key elements. First, our model consists of a separate latent space that consists of a content space z_c and a style space z_s as illustrated in Fig. 1. The former is encoded with content of the songs such as the note pitches used and its duration, while the latter is to embed abstract information of the compositional style. Secondly, and crucial to guide our model to generate songs of a specific style, is the proposal of continuous style embeddings, in contrast to discrete labels. Discrete labels are usually one-hot vectors fed to the model at generation time as conditions, in our solution, we employ a continuous embedding instead.

Empirically, we show the ability of our model to generate music with a specific style, in comparison with conventional solutions (Table 2 and Fig. 5). Also, we explain from a musical theory point of view that our generated music mimics closely to the conditioned style (Fig. 3). Nonetheless, we demonstrate the ability of our model to perform musical style

978-1-7281-1331-9/20/$31.00

transfer (Table 3-4, Fig. 6). Due to nature of this work, we encourage readers to listen the audio samples provided here[1].

2. RELATED WORK

Generative models have always been a field of interest for researchers, but the general formulations of VAE and GAN do not have much interactive properties that allow users to control the content or attributes of the generated output. Recently, [1, 6, 7, 12, 13] extend the original GAN idea and develop new formulations that provide users a way to control the content of the output. Technically, there are two ways to train a model that allows user control. Formerly, semantic labels are provided to the model during training and they are used as conditions at generation time. These models are trained in either supervised or semi-supervised mode. While the latter is to train a model to perform unsupervised clustering on the data. At generation time, new samples are generated by sampling from different clusters. Our work falls on the former.

Earlier works in symbolic music generation are usually based on"next step prediction", where a priming melody is given as a prior and the model generates a continuation of it [3, 14]. Due to this, the style of the output music is limited by the training dataset. For instance, [15, 16] can only generate music in the style of Bach, and not others as they were trained with Bach's chorales. In contrast, our work is able to provide users the ability and flexibility to condition on any musical style for their generated music.

Based on our knowledge, recent advances have also shown limited success in allowing user the flexibility to control over the style of generated music. In [17], the authors presented their VAE model that is able to generate music conditioned on attributes such as note density, diatonic scale and note syncopations. These conditions are extracted as attribute vectors after training. Payne [18] prepended composer and instrumentation tokens to the samples before training. The tokens are then used at generation time to condition the generated music. Simon et al. [11] extended the idea from [17] and proposed a model that can generate music conditioned on chord progressions. As a summary, these aforementioned works are fairly limited since it only allows the control over musical attributes such as rhythmic patterns and chord progressions. The idea of conditioning on the musical compositional style as to our proposed work is still less explored.

Related most closely to our work is MIDI-VAE [19], where a VAE model was trained to model the dynamics and instrumentation of different genres of music. The model can then perform neural style transfer between different genres of music, for example from classical to jazz. Our work resembles the framework suggested in [19], however, our work modifies this framework considerably where we convert the softmax style latent space into a continuous style embedding.

[1] https://bit.ly/3b5QYKW

3. METHODOLOGY

3.1. Data Representation

The dataset we used is built from MIDI files. We extended the idea of [10, 20] and encoded the MIDI event messages in a list of vectors. Each sample is split into a segment of 4 bars, and each of this segment is a list of notes represented in vector X ordered by note start times and in ascending pitch order for chords. Each vector X contains 3 features: (P, dt, D). P is a one-hot tensor $P \epsilon \{0,1\}^{n_T,89}$, where n_T is the number of notes in the segment and 89 is the total number of distinct pitches plus an additional silent pitch to represent the end of a 4 bar phrase. dt is the amount of time in beats after the previous note is played and before the next note is played. We quantized a whole note (4 beats) into 33 bins and represent dt as a one-hot tensor $dt \epsilon \{0,1\}^{n_T,33}$, where dt=0 means the note is played together with the previous note, which represents a chord. D is the duration in beats of a note being held, and it is encoded similar to dt, $D \epsilon \{0,1\}^{n_T,33}$. The quantization of timing information into 33 bins allows us to model all common note lengths in musical theory (semibreve, minim, crotchet, etc. and their dotted counterpart) and also complex note lengths like ornaments and triplets.

3.2. Model Architecture

3.2.1. VAE Revisited

Our model adopted the sequence-to-sequence VAE [17, 19] that consists of two networks, namely the encoder and decoder. Both the encoder and decoder are modeled as variants of recurrent neural networks (RNN). The VAE imposes a prior distribution $p(z)$ on the latent variables z, while the encoder learns an approximated posterior $q(z|x) = \mathcal{N}(\mu, \sigma)$. Due to the sampling process, the gradients of this network is intractable. As such, the reparametrization trick is used, where $z = \mu + \epsilon\sigma$. μ and σ are the mean and standard deviation of the latent distribution, both are modeled by the encoder. ϵ is a random Gaussian noise. The decoder $p(x|z)$ learns to reconstruct the input. The general formulation of a VAE is as follows:

$$L_V = L_r - \beta D_{KL}(q(z|x)||p(z)) \quad (1)$$

where L_r is the reconstruction loss, which in our case it is the cross-entropy between the reconstructed output $\tilde{X}$ and X. The second term is the Kullback-Leibler (KL) divergence between the posterior and prior distribution. Optimizing the KL term will force the latent distribution to be closed to a Gaussian distribution. The β term is a weight to balance the tradeoffs between the reconstruction and KL term.

3.2.2. Proposed Method

Our proposed model is depicted in Fig. 1, where the dotted lines path represents a vanilla VAE for reference. To learn

the style of a music, in our proposed architecture, our latent space z is split into two parts, z_s and z_c. The idea behind this is that the encoder encodes the "content" of an input music, such as note pitches and note lengths into z_c, while z_s will be optimized to contain the "style" information that guides the decoder to generate a music in a certain style.

To obtain z_s, our encoder first generates a one-hot categorical variable $z_{cat} \, \epsilon \, \{0,1\}^s$, where s denotes the number of styles in the dataset. Due to the nature of discrete variables, the gradients of z_{cat} is intractable. For this, we adopt the Gumbel distribution trick [21] formulated as:

$$G = -\log(-\log(Unif[0,1])) \tag{2}$$

$$z_{cat} = softmax((\alpha + G)/\tau_{gumbel}) \tag{3}$$

where G is the Gumbel noise, α is the logits for z_{cat} from the encoder and τ_{gumbel} is the temperature. The temperature is a hyperparameter, as it approaches zero, z_{cat} becomes a one-hot vector.

We then introduce a learnable style codebook that randomly initialize at the start of training. The codebook is used to retrieve style embeddings based on z_{cat}. Our codebook consists of s embeddings, each with a dimension of dim_s, which gives it a size of $s \times dim_s$. To obtain z_s, we perform a matrix multiplication between z_{cat} and the style codebook:

$$z_s = z_{cat} \otimes style\_codebook \tag{4}$$

z_s will then has the shape of dim_s. Thus, we can ensure that different songs of the same style will have the same style embeddings z_s. Also, in order to ensure our model learns the style correctly, z_{cat} is optimized with style labels y using cross-entropy:

$$L_s = -\sum_{s=1}^{S} y \log z_{cat} \tag{5}$$

where s is the number of styles in the dataset.

Our z_c is similar as to the original latent representation $z = \mu + \epsilon\sigma$ described in Section 3.2.1. Then, z_s and z_c are concatenated to form z'. Finally, z' is passed through a linear layer to form the initial state of the decoder RNN.

Posterior collapse is an issue where the decoder ignores the latent vectors, which in our case is z_c, resulting in the KL term dropping to zero and rendering z_c useless. This issue is common in a seq2seq VAE setting due to the nature of the autoregressive decoder [22]. For this, we adopted μ-forcing [23] which adds a regularizing term L_μ to the VAE formulation:

$$L_\mu = max(0, \beta_\mu - \frac{1}{2N}\sum_{n=1}^{N}(\mu^n - \bar{\mu})^T(\mu^n - \bar{\mu})) \tag{6}$$

where β_μ is a margin and N is the batch size. μ is the mean vector modeled by the encoder to parameterize the distribution of z_c. The term L_μ forces the sample variance of μ to be controlled on the level of β_μ which maintains the mutual information of X and z'.

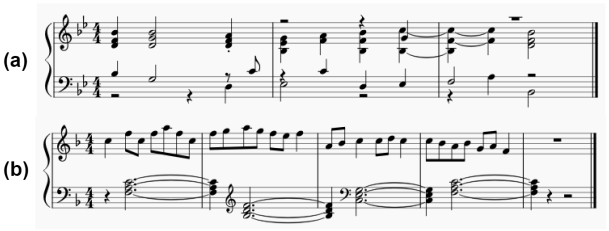

Fig. 2. Samples from (a) JSB and (b) NMD public datasets.

As a whole, the final formulation of our proposed model is as follows:

$$L'_V = L_r - \beta D_{KL}(q(z_c|x)||p(z_c)) + L_s + L_\mu \tag{7}$$

At generation time, we can specify z_{cat} according to the wanted style, sample z_c from a Gaussian distribution and feed them to the decoder to generate a new song.

4. IMPLEMENTATION DETAILS

4.1. Dataset & Pre-Processing

In this paper, we employ two public music datasets that have been studied heavily: Bach Chorales (JSB) dataset and Nottingham Music Database (NMD). The JSB dataset contains 370 four-part chorales harmonized by Johann Sebastian Bach while NMD contains over 1200 American and European folk tunes. Both JSB and NMD have very distinct style, and samples are depicted in Fig. 2. The JSB chorales are written in four parts: soprano, alto, tenor and bass. As can be seen in Figure 2(a), the music is formed by chords of four notes, with passing notes in between. In contrast, the folk tunes in the NMD consist of a simple monophonic melody line, accompanied by a chord sequence. The data are converted into the representation described in Section 3.1, split into training and testing sets and each song is split into non-overlapping segments of four bars.

4.2. Model Hyperparameters

Similar to [24], our encoder is a bidirectional HyperLSTM [25] layer and the decoder is a unidirectional HyperLSTM layer. Although LSTM is a more popular choice, in our experiments, we show that in Table 1 the HyperLSTM cell has a better performance when handling long, complex sequences like music. The LSTM model seems to overfit the data significant earlier as opposed to the HyperLSTM model during the training phase.

The style codebook dimension is set to 80 and the dimension of z_c is set to 120. We applied dropout to both the encoder and decoder with rates of 0.5 and 0.2 respectively. β_μ

Model	pitch, P	dt	duration, D	KL
LSTM	0.77	0.94	0.81	0.94
Proposed$_{xs}$	0.21	0.63	0.45	1.10
Proposed	0.82	0.96	0.83	1.13

Table 1. The reconstruction accuracy and KL divergence on the test set of models of various settings.

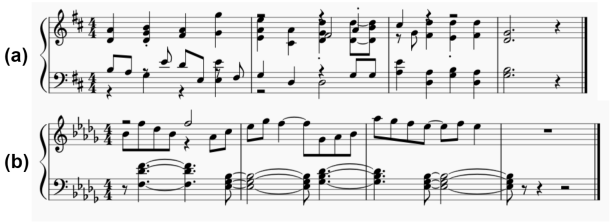

Fig. 3. Generated music samples by the proposed model in the style of (a) JSB and (b) NMD.

is set at 1.3 and KL weight β is annealed from 0 to 0.8. Adam optimizer is used with a learning rate of 5e-4. All these hyperparameters are set empirically, and trained for 400 epochs.

5. EXPERIMENTAL RESULTS & DISCUSSION

5.1. Reconstruction Performance

Table 1 presents the accuracy of the test set reconstruction and KL divergence of our proposed model in various settings. The LSTM model has the same settings as the proposed model except that the RNN cells used in both the encoder and decoder are LSTM instead of HyperLSTM. It can be seen that the proposed model outperforms the LSTM model, especially in reconstructing the pitch. Furthermore, the proposed model is able to better utilize z_c to reconstruct the input, as justified by the significant higher KL term compared to the LSTM model. To show the importance of the proposed z_s, we replaced z_s in the proposed model with all zeros and then reconstruct the test set. This model is denoted as proposed$_{xs}$ in Table 1. The poor reconstruction performance proves that z_s is indeed important for the proposed model to reconstruct the input or generate new music.

5.2. Style-conditioned Music Generation

Unlike most of the previous works that work on conditional generative modeling [6, 12, 19], we did not feed the discrete style representation z_{cat} directly to the decoder. Instead, we argue that a one-hot categorical vector does not sufficient information to guide the model to generate a musical sequence of a specific style. Thus, we proposed a continuous embedding using our "style codebook" described in Section 3.2. To

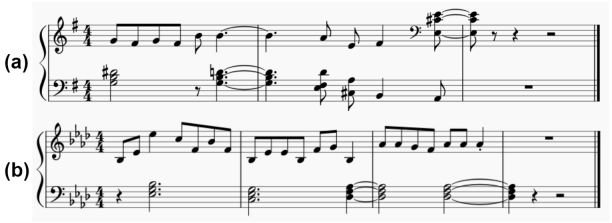

Fig. 4. Generated samples by the baseline model in the style of (a) JSB and (b) NMD.

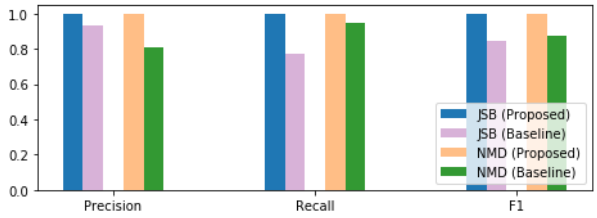

Fig. 5. Comparison of the encoder classification scores on musical samples generated by proposed and baseline models.

justify this, we trained a baseline model very similar to MidiVAE [19] that concatenates z_{cat} directly to z_c.

In this experiment, we first generate 500 music samples of each style by sampling z_c and feed in the respective z_{cat} to the model. Using the encoder as a classifier, we feed the generated output into it and the output z_{cat} is taken as the classified style. Table 5 shows the precision and recall score for the classification of each style of both models. The baseline model scored lower than the proposed model. A significant amount of JSB samples generated by the baseline model (Fig. 4 shows a sample) were in the style of NMD, thus the lower precision score on the NMD side. In contrast, our proposed model can generate songs of either style with ease, samples are depicted in Fig. 3. Note that the encoder is able to classify the style of an input song with perfect accuracy as presented in Table 3.

Besides investigating the style of the generated samples using a neural network, we conducted a simple qualitative analysis as well. As illustrated in Fig. 2, from a musical theory point of view, due to the four-part writting style, the JSB dataset contains significantly more chords per segment in contrast to the NMD dataset. In addition to that, songs in the style of JSB will have a wider pitch range and thus will contain more unique pitches per segment. In the top half of Table 2, we compared the mean of the number of chords and number of unique pitches per segment in the training dataset (namely as "Training") against the generated samples. It is shown that the proposed model outperforms the baseline as the values are much closer to the training dataset.

Samples	# of chords		# of unique pitch	
	JSB	NMD	JSB	NMD
Training	16.20	5.13	20.10	13.83
Baseline	6.82	4.68	15.20	**15.63**
Proposed	**13.94**	**4.94**	**21.48**	16.68
$\text{Baseline}_{\text{transfer}}$	5.03	15.46	13.55	19.20
$\text{Proposed}_{\text{transfer}}$	**13.84**	**5.33**	**19.12**	**16.29**

Table 2. Comparing the mean number of chords and number of unique pitches per segment of each style to the generated samples and style transfer samples for each model. The style transfer results are indicated by the subscript "transfer". Value that is closer to "Training" is better.

Model	Before	After
Baseline	1.0	0.95
Proposed	1.0	0.13

Table 3. A comparison of the accuracy of style prediction before and after performing style transfer.

5.3. Style Transfer

In addition to music generation, we also tested the ability of our proposed model to perform style transfer. In this case, we reconstructed each song in the testing set with the different z_{cat} and the results are presented in the bottom part of Table 2. It can be seen that even after style transfer, the samples of the baseline model still retain the characteristics of the original style. On the other hand, the samples of the proposed model have the characteristics of the transferred style. Fig. 6 shows the style transferred samples. Through a manual inspection of the style transferred samples from the proposed model, we observed that the key and the accidentals of the samples are preserved after the style transfer. Although the note pitches are preserved, it may be either reordered or transpose to a higher or a lower octave and the duration may change as well, in order to fit the target style. This observation proves that the information of note pitches are stored in z_c.

Table 3 shows the accuracy of the encoder predicting the style of the songs in the testing set before and after style transfer. It can be seen that after the style transfer, the encoder performed poorly in classifying the samples generated by the proposed model, indicating a successful style transfer. In contrast, the encoder classifies the style transferred samples of the baseline model to its original style.

5.4. Style Information in $\mathbf{z_c}$

Disentangling features from the latent representation is one of the popular method in the conditional generative modeling literature [26–28]. We did not explicitly design the proposed model in any way that forces it to only encode the style information in the style embeddings and the remaining contents in z_c. Thus, it is still possible for z_c to contain style information. To investigate this, we trained a logistic regression classifier on z_c with y as target. The classification accuracy on the testing set is presented in Table 4. It shows that z_c of the baseline model is more distinctive comparing to the proposed model as indicated by the higher accuracy. The overall results show that style information is still encoded in z_c. The latent space plot is presented in Fig. 7 and qualitatively, it can be seen that the separation of clusters for JSB and NMD is quite obvious.

Model	Before	After
Baseline	0.97	0.94
Proposed	0.86	0.71

Table 4. Classification accuracy of style based on z_c. Results are presented for samples of both models before and after style transfer.

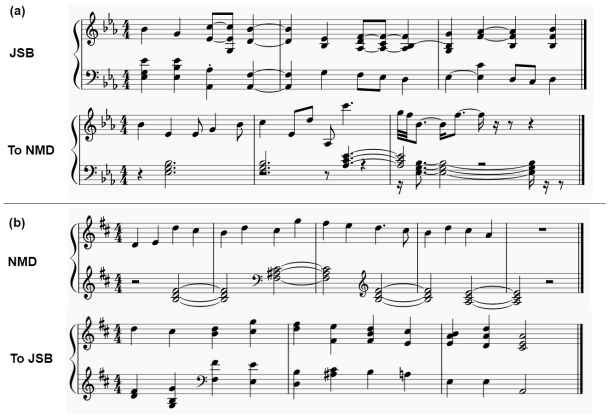

Fig. 6. Samples of generated song after style transfer. (a) Style transfer from JSB to NMD. (b) Style transfer from NMD to JSB.

6. CONCLUSION

In this work, we proposed a modification to the vanilla sequence-to-sequence variational autoencoder that allows user to condition the style of generated output music. Our model involves a continuous style embedding for each style in the dataset. Through our experiments, we showed that our proposed method outperforms the baseline which directly feeds discrete style labels to the model similar to other works in the literature. In future work, we aim to explore our proposed method on other kinds of sequential data.

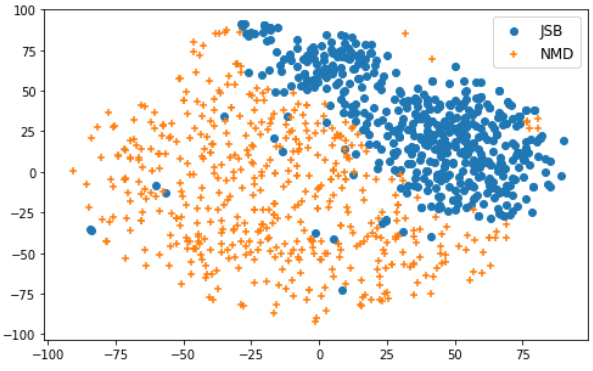

Fig. 7. Latent space plot of z_c in 2D with t-SNE.

7. REFERENCES

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *ICLR*, 2019.

[2] Bei Liu, Jianlong Fu, Makoto P Kato, and Masatoshi Yoshikawa, "Beyond narrative description: Generating poetry from images by multi-adversarial training," in *ACM-MM*, 2018.

[3] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan, "This time with feeling: Learning expressive musical performance," *CoRR*, 2018.

[4] "Magenta deeplocal the flaming lips = fruit genie," 2019.

[5] "Yacht's new album is powered by ml artists," 2019.

[6] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *NIPS*, 2016.

[7] Tero Karras, Samuli Laine, and Timo Aila, "A style-based generator architecture for generative adversarial networks," in *CVPR*, 2019.

[8] 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," in *ICLR*, 2019.

[9] 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 *AAAI*, 2018.

[10] Chris Donahue, Ian Simon, and Sander Dieleman, "Piano genie," in *ACM IUI*, 2019.

[11] 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.

[12] Emilien Dupont, "Learning disentangled joint continuous and discrete representations," in *NeurIPS*, 2018.

[13] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *CVPR*, 2018.

[14] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in *ICML*, 2012.

[15] Gaëtan Hadjeres, François Pachet, and Frank Nielsen, "Deepbach: a steerable model for bach chorales generation," in *ICML*, 2017.

[16] Feynman T Liang, Mark Gotham, Matthew Johnson, and Jamie Shotton, "Automatic stylistic composition of bach chorales with deep lstm.," in *ISMIR*, 2017.

[17] 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 *ICML*, 2018.

[18] Christine Payne, "Musenet," Oct 2019.

[19] Gino Brunner, Andres Konrad, Yuyi Wang, and Roger Wattenhofer, "MIDI-VAE: modeling dynamics and instrumentation of music with applications to style transfer," in *ISMIR*, 2018.

[20] Florian Colombo and Wulfram Gerstner, "Bachprop: Learning to compose music in multiple styles," *arXiv preprint arXiv:1802.05162*, 2018.

[21] Eric Jang, Shixiang Gu, and Ben Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[22] 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.

[23] Dayiheng Liu, Xu Yang, Feng He, Yuanyuan Chen, and Jiancheng Lv, "mu-forcing: Training variational recurrent autoencoders for text generation," *arXiv preprint arXiv:1905.10072*, 2019.

[24] David Ha and Douglas Eck, "A neural representation of sketch drawings," in *ICLR*, 2018.

[25] David Ha, Andrew Dai, and Quoc V Le, "Hypernetworks," *arXiv preprint arXiv:1609.09106*, 2016.

[26] Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel, "Variational lossy autoencoder," *arXiv preprint arXiv:1611.02731*, 2016.

[27] Yingzhen Li and Stephan Mandt, "Disentangled sequential autoencoder," *arXiv preprint arXiv:1803.02991*, 2018.

[28] Irina Higgins, Loic 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.," *ICLR*, 2017.