Understanding Variational Autoencoders

Mauricio Araneda

September 22, 2021

Introduction

- We are interested on generating new data after observing a given dataset.
- Deep learning based generative models have gained more and more interest for this task.
- Today we are going to focus on generating new data through Variational Autoencoders (VAEs).

What is a VAE?

- What is Dimensionality Reduction?
- What is an Autoencoder?
- What is the latent space and why regularising it?
- What is the link between VAEs and variational inference?
- How to generate new data from VAEs?
- Where can I use VAEs?

Dimensionality Reduction

- In machine learning, **dimensionality reduction** is the process of reducing the number of features that describe some data.
- Dimensionality reduction can then be interpreted as **data compression** where the encoder compress the data (from the initial space to the encoded space, also called **latent space**) whereas the decoder decompress them.
- Of course, depending on the initial data distribution, the latent space dimension and the encoder definition, this compression can be **lossy**, meaning that a part of the information is lost during the encoding process and **cannot be recovered when decoding**.

Dimensionality Reduction

- The main purpose of a dimensionality reduction method is to find the best encoder/decoder pair among a given family.
- In other words, for a given set of possible encoders and decoders, we are looking for the pair that keeps the maximum of information when encoding and, so, has the minimum of reconstruction error when decoding.

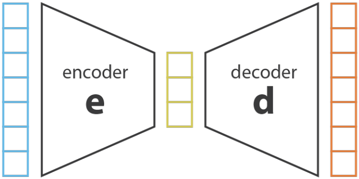
Dimensionality Reduction Problem

- If we denote respectively E and D the families of encoders and decoders we are considering, then the dimensionality reduction problem can be written:

$$(e^*, d^*) = \underset{(e,d) \in ExD}{\text{argmin}}\ \epsilon(x, d(e(x)))$$

where $\epsilon(x, d(e(x)))$ defines the reconstruction error measure between the input data x and the encoded-decoded data $d(e(x))$

Dimensionality Reduction



x — **initial data** in space R^n

$e(x)$ — **encoded data** in latent space R^m (with $m<n$)

$d(e(x))$ — **encoded-decoded data** back in the initial space R^n

$x = d(e(x))$ → **lossless encoding** no information is lost when reducing the number of dimensions

$x \neq d(e(x))$ → **lossy encoding** some information is lost when reducing the number of dimensions and can't be recovered later

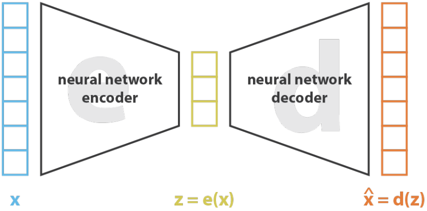
Figure: Illustration of the dimensionality reduction principle with encoder and decoder.

Autoencoders (AE)

- The general idea of autoencoders is pretty simple and consists on estimating the encoder function $e(x)$ and decoder function $d(x)$ through neural networks.
- Neural networks will, in theory, learn the best encoding-decoding scheme using an iterative optimisation process.
- The problem we are aiming to minimize is:

$$(\theta_e^*, \theta_d^*) = \underset{\theta_e \theta_d}{\mathrm{argmin}} \epsilon(x, d(e(x)))$$

Autoencoders (AE)



$$\text{loss} = \| x - \hat{x} \|^2 = \| x - d(z) \|^2 = \| x - d(e(x)) \|^2$$

Figure: Illustration of an autoencoder with its loss function.

Autoencoders (AE)

- The more complex the architecture is, the more the autoencoder can proceed to a high dimensionality reduction while keeping reconstruction loss low.
- If our encoder and our decoder have enough degrees of freedom, we can reduce any initial dimensionality to 1, with no loss during the process.
- However, an important dimensionality reduction with no reconstruction loss often comes with a price:
  - Lack of interpretable and exploitable structures in the latent space (lack of regularity)
  - Loss of the major part of the data structure information in the reduced representations.

  .
- For these two reasons, the dimension of the latent space and the depth of autoencoders have to be controlled and adjusted depending on the final purpose of the dimensionality reduction.

Autoencoders (AE)

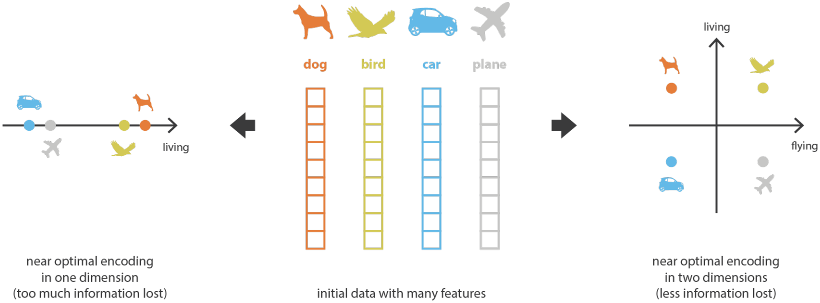


Figure: When reducing dimensionality, we want to keep the main structure there exists among the data.

Limitations of autoencoders for content generation

- Once the autoencoder has been trained, we have both the encoder and a decoder functions. From there our focus will be on the latent space.
- The idea is to take points from the latent space and decode them to get new contents. Ideally, closer points on this space will be closer semantically.
- However, there is no way to ensure that the encoder will organize the latent space in a smart way compatible with the generative process we just described.
- The regularity of the latent space for autoencoders is a difficult point that depends on the distribution of the data in the initial space, the dimension of the latent space and the architecture of the encoder.

Limitations of autoencoders for content generation

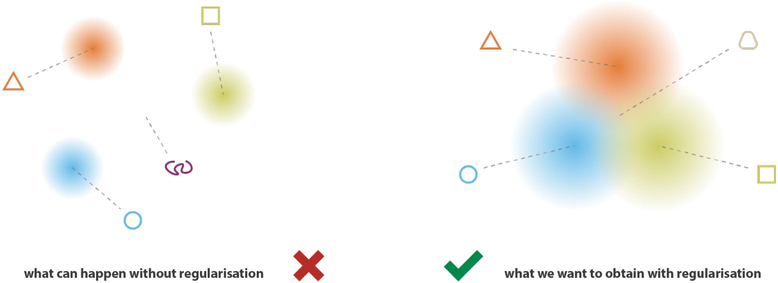

what can happen without regularisation
✗
✓
what we want to obtain with regularisation

Figure: We can generate new data by decoding points that are randomly sampled from the latent space. The quality and relevance of generated data depend on the regularity of the latent space.

Variational Autoencoder (VAE)

- A VAE is an autoencoder whose encodings distribution is regularised during the training in order to ensure that its latent space has good properties allowing us to generate new data.

Probabilistic Framework

- In order to introduce some regularisation of the latent space, we proceed to a slight modification of the encoding-decoding process: instead of encoding an input as a single point, we encode it as a distribution over the latent space.
- Let's define a graphical probabilistic model to describe our data. We denote by x the variable that represents our data and assume that x is generated from a latent variable z (the encoded representation) that is not directly observed.
- Thus, for each data point, we are assuming they were generated through:
  1. A latent variable z sampled from the prior distribution $p(z)$.
  2. The data x is sampled from $p(x|z)$.

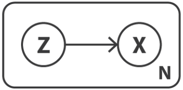


Figure: Graphical model of the data generation process..

Probabilistic Framework

- With this approach we can note that we are introducing a ”probabilistic decoder” defined by $p(x|z)$.
- We can then build our distributions for $p(x|z)$ and $p(z)$ in such a way they both satisfy our regularisation requirements.
- Following the ”probabilistic decoder” idea, we can define the ”probabilistic encoder” as $p(z|x)$.
- Bayes theorem makes the link between the prior $p(z)$, the likelihood $p(x|z)$, and the posterior $p(z|x)$ so we can calculate the ”encoder”.

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x|z)p(z)}{\int p(x|u)p(u)du}$$

- The problem now is how to deal with the denominator...

Probabilistic Framework

- Let's define $p(z)$ and $p(x|z)$

$$p(z) \equiv \mathcal{N}(0, \mathcal{I})$$

$$p(x|z) \equiv \mathcal{N}(f(z), c\mathcal{I}) f \in \mathcal{F} c > 0$$

- Let's consider, for now, that f is well defined and fixed. In theory, as we know $p(z)$ and $p(x|z)$, we can use the Bayes theorem to compute $p(z|x)$.

Variational Inference Formulation

- In statistics, variational inference (VI) is a technique to approximate complex distributions.
- We want to **estimate the Posterior Distribution** ($p(z|x)$) given our likelihood ($p(x|z)$) and prior ($p(z)$). [*Inference*]
- We want to **optimize** over a **parametrized family** of functions in order to look for the best approximation of our target distribution. [*Variational Calculus*]
- The best element in the family is one that minimise a given approximation error measurement, in this case Kullback-Leibler divergence.

$$KL(p, q) = \mathbb{E}_{z \sim p}[\log p(z)] - \mathbb{E}_{z \sim p}[\log q(z)]$$

Variational Inference Formulation

- Here we are going to approximate $p(z|x)$ by a Gaussian distribution $q_x(z)$ whose mean and covariance are defined by two functions, g and h, of the parameter x.

$$q_x(z) \equiv \mathcal{N}(g(x), h(x)) g \in \mathcal{G} h \in \mathcal{H}$$

- So, we have defined this way a family of candidates for variational inference and need now to find the best approximation among this family by optimising the functions g and h (in fact, their parameters) to minimise the Kullback-Leibler divergence between the approximation and the target $p(z|x)$.

Variational Inference Formulation

- In other words, we are looking for the optimal g* and h* such that

$$
\begin{aligned}
(g^*, h^*) &= \underset{(g,h)\in\mathcal{G}x\mathcal{H}}{\operatorname{argmin}}\ KL(q_x(z), p(z|x)) \\
&= \underset{(g,h)\in\mathcal{G}x\mathcal{H}}{\operatorname{argmin}}\ (\mathbb{E}_{z\sim q_x}(\log q_x(z)) - \mathbb{E}_{z\sim q_x}(\log \frac{p(x|z)p(z)}{p(x)}) \\
&= \underset{(g,h)\in\mathcal{G}x\mathcal{H}}{\operatorname{argmin}}\ \Big(\mathbb{E}_{z\sim q_x}(\log q_x(z)) - \mathbb{E}_{z\sim q_x}(\log p(z)) - \mathbb{E}_{z\sim q_x}(\log p(x|z)) \\
&- \mathbb{E}_{z\sim q_x}(\log p(x))\Big) \\
&= \underset{(g,h)\in\mathcal{G}x\mathcal{H}}{\operatorname{argmax}}\ (\mathbb{E}_{z\sim q_x}(\log p(x|z)) - KL(q_x(z), p(z)))
\end{aligned}
$$

Variational Inference Formulation

- The last equation is super important. It is called *Evidence Lower Bound (ELBO)* and it reflects a tradeoff of two quatities:
    1. Maximising the likelihood of the "observations". Can be seen as reconstruction quality over x.
    2. Staying close to the prior distribution. Can be seen as the degree of compact representation on the latent space.
- We have assumed that $p(x|z)$ is defined through the function f which we assume is known and fixed and we have showed that, under such assumptions, we can approximate the posterior $p(z|x)$ using variational inference technique. However, in practice this function f, that defines the decoder, is not known and also need to be chosen.

Variational Inference Formulation

- We want to choose the function f that maximises the expected log-likelihood of x given z when z is sampled from $q_x^*(z)$.
- In other words, for a given input x, we want to maximise the probability to have $\hat{x} = x$ when we sample z from the distribution $q_x^*(z)$ and then sample $\hat{x}$ from the distribution $p(x|z)$.
- Thus, we are looking for the optimal f^* such that:

$$
\begin{aligned}
f^* &= \underset{f \in F}{\operatorname{argmax}} \mathbb{E}_{z \sim q_x^*}(\log p(x|z)) \\
&= \underset{f \in F}{argmax} \mathbb{E}_{z \sim q_x^*}\left(- \frac{||x - f(z)||^2}{2c} \right)
\end{aligned}
$$

Variational Inference Formulation

- Gathering all the pieces together, we are looking for optimal f^*, g^*, h^* such that

$$(f^*, g^*, h^*) = \underset{(f,g,h)\in FxGxH}{argmax} \left(\mathbb{E}_{z\sim q_x} \left(-\frac{||x - f(z)||^2}{2c} \right) - KL(q_x(z), p(z)) \right) \quad (1)$$

- We can identify in this objective function: the reconstruction error between x and $f(z)$ and the regularisation term given by the KL divergence between $q_x(z)$ and $p(z)$ (which is a standard Gaussian).
- We can also notice the constant c that rules the balance between the two previous terms. The higher c is the more we assume a high variance around $f(z)$ for the probabilistic decoder in our model and, so, the more we favour the regularisation term over the reconstruction term.

Bringing Neural Networks into the model (Encoder)

- In practice, g and h are not defined by two completely independent networks but share a part of their architecture and their weights so that we have:

$$g(x) = g_2(g_1(x))$$

$$h(x) = h_2(h_1(x))$$

$$g_1(x) = h_1(x)$$

Bringing Neural Networks into the model (Encoder)

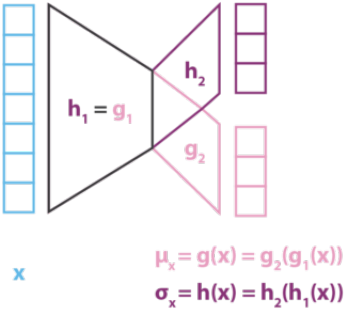


Figure: Encoder part of the VAE.

Bringing Neural Networks into the model (Decoder)

- Contrarily to the encoder part that models $p(z|x)$ and for which we considered a Gaussian with both mean and covariance that are functions of x (g and h), our model assumes for $p(x|z)$ a Gaussian with fixed covariance. The function f of the variable z defining the mean of that Gaussian is modelled by a neural network and can be represented as follows:

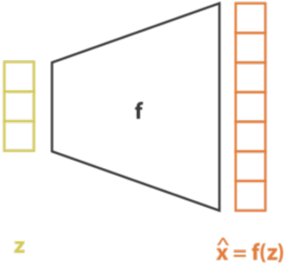


Figure: Decoder part of the VAE.

Bringing Neural Networks into the model

- The overall architecture is then obtained by concatenating the encoder and the decoder parts.
- **The sampling process has to be expressed in a way that allows the error to be backpropagated** through the network. We use a reparametrisation trick, to make the gradient descent possible despite the random sampling.
- We want z to be described with determistic parameters to backpropagate the gradients, so we will describe it through a third variable ζ which will introduce the undeterministic sampling process, then z can be expressed as:

$$z = h(x)\zeta + g(x), \zeta \sim \mathcal{N}(0, I)$$

Bringing Neural Networks into the model

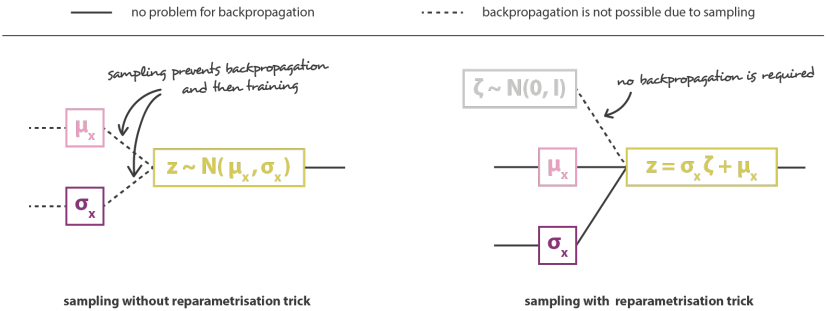
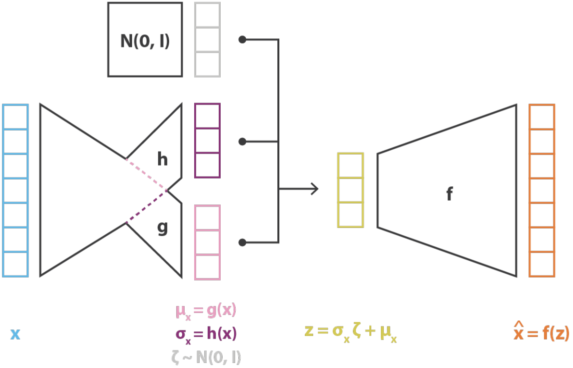

no problem for backpropagation
backpropagation is not possible due to sampling
sampling prevents backpropagation and then training
μ_x
σ_x
$z \sim N(\mu_x, \sigma_x)$
$\zeta \sim N(0, I)$
no backpropagation is required
μ_x
σ_x
$z = \sigma_x \zeta + \mu_x$
sampling without reparametrisation trick
sampling with reparametrisation trick

Figure: Illustration of the reparametrisation trick.

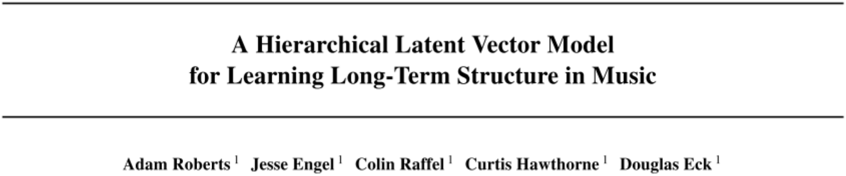
Bringing it all together



$$\text{loss} = C \,||\, x - \hat{x} \,||^2 + KL[\, N(\mu_x, \sigma_x), N(0, I) \,] = C \,||\, x - f(z) \,||^2 + KL[\, N(g(x)\,, h(x)), N(0, I) \,]$$

Figure: Variational Autoencoders representation.

VAE Applications (MusicVAE)



A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music

Adam Roberts[1] Jesse Engel[1] Colin Raffel[1] Curtis Hawthorne[1] Douglas Eck[1]

VAE Applications (MusicVAE)

- Previous work attempted to capture the temporal dependencie on data through the use of RNN (*Recurrent VAEs*).
- This approach had two major drawbacks:
  1. RNNs were powerful enough to encode all the data, leading to a disregard of the latent code [*Posterior Collapse*]. This could be interpreted in the ELBO as a maximization only on the reconstruction quality term while setting the regularization term to zero.

     $$\mathcal{L} = \mathbb{E}[\log_{p_\theta}(x|z) - KL(q_\lambda(z|x)||p(z))]$$

  2. The model must compress the entire sequence to a single latent vector. This approach can work for short sequences, but it begins to fail as the sequence length increases.
- This paper presents a hierarchical RNN for the decoder, which limits the scope of the decoder to force it to use the latent code to model long-term structure.

VAE Applications (MusicVAE)

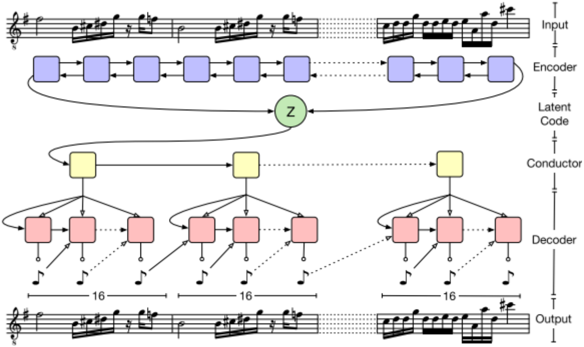
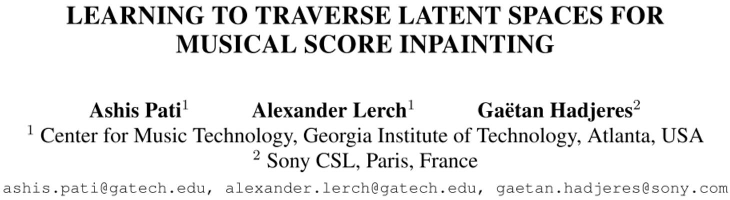


Figure: Schematic of the hierarchical recurrent Variational Autoencoder model, MusicVAE.

VAE Applications (Music InpaintNet)



LEARNING TO TRAVERSE LATENT SPACES FOR MUSICAL SCORE INPAINTING

Ashis Pati[1] **Alexander Lerch**[1] **Gaëtan Hadjeres**[2]
[1] Center for Music Technology, Georgia Institute of Technology, Atlanta, USA
[2] Sony CSL, Paris, France
ashis.pati@gatech.edu, alexander.lerch@gatech.edu, gaetan.hadjeres@sony.com

VAE Applications (Music InpaintNet)

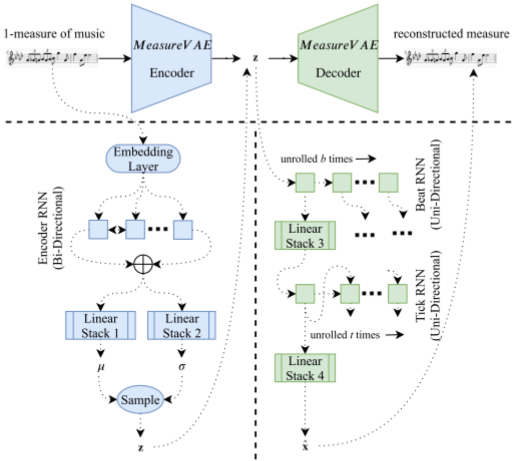


Figure: MeasureVAE schematic. Individual components of the encoder and decoder are shown below the main blocks (dotted arrows indicate data flow within the individual components). z denotes the latent vector and ˆx denotes the reconstructed measure.

VAE Applications (Music InpaintNet)

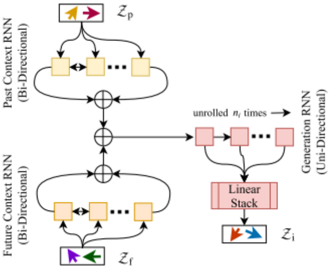
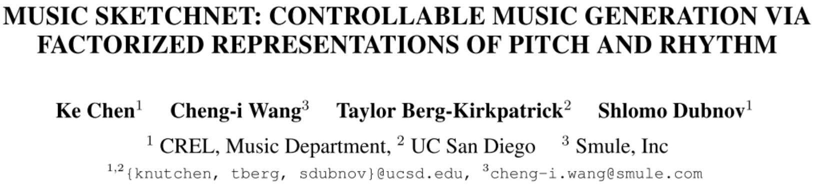


Figure: LatentRNN schematic. The Past-Context and Future-Context-RNNs encode Z_p and Z_f, respectively. The Generation-RNN initialized using a concatenation of context-RNNs embeddings is unrolled n_i times to get Z_i.

VAE Applications (Music SketchNet)



MUSIC SKETCHNET: CONTROLLABLE MUSIC GENERATION VIA FACTORIZED REPRESENTATIONS OF PITCH AND RHYTHM

Ke Chen[1] **Cheng-i Wang**[3] **Taylor Berg-Kirkpatrick**[2] **Shlomo Dubnov**[1]

[1] CREL, Music Department, [2] UC San Diego [3] Smule, Inc

[1,2]{knutchen, tberg, sdubnov}@ucsd.edu, [3]cheng-i.wang@smule.com

VAE Applications (Music SketchNet)

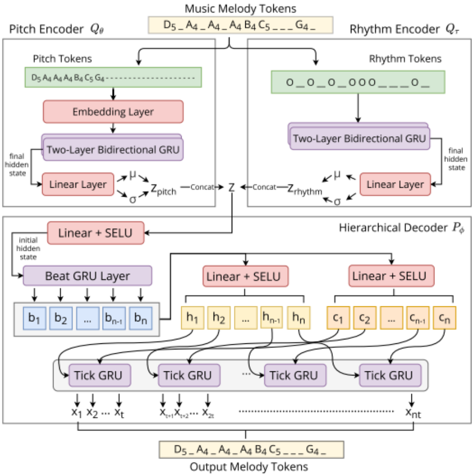
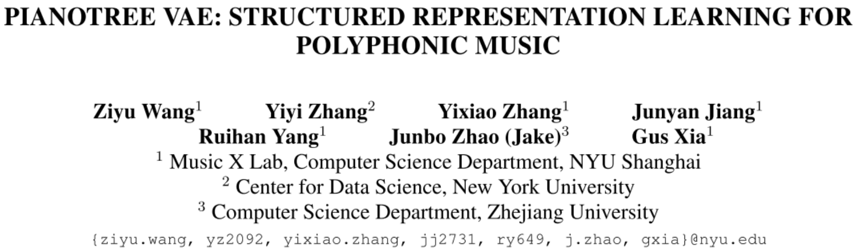


Figure: SketchVAE structure: pitch encoder, rhythm encoder and hierarchical decoder. Rhythm tokens: the upper dashes denote the onsets of note, and the bottom dashes denote the hold/duration state. Pitch symbols represent the tokens numbers for better illustration.

VAE Applications (PianoTree VAE)



PIANOTREE VAE: STRUCTURED REPRESENTATION LEARNING FOR POLYPHONIC MUSIC

Ziyu Wang[1] **Yiyi Zhang**[2] **Yixiao Zhang**[1] **Junyan Jiang**[1]
Ruihan Yang[1] **Junbo Zhao (Jake)**[3] **Gus Xia**[1]
[1] Music X Lab, Computer Science Department, NYU Shanghai
[2] Center for Data Science, New York University
[3] Computer Science Department, Zhejiang University
{ziyu.wang, yz2092, yixiao.zhang, jj2731, ry649, j.zhao, gxia}@nyu.edu

VAE Applications (PianoTree VAE)

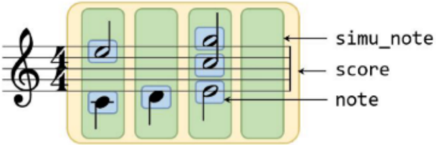


Figure: Illustration of the proposed polyphonic syntax.

VAE Applications (PianoTree VAE)

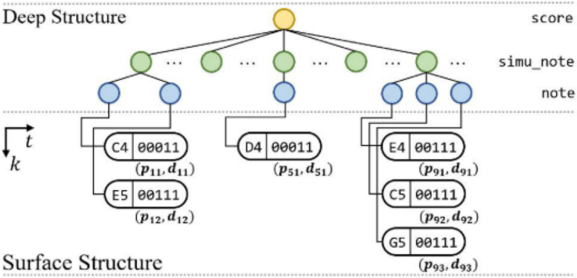


Figure: An illustration of PianoTree data structure to encode the music example above.

VAE Applications (PianoTree VAE)

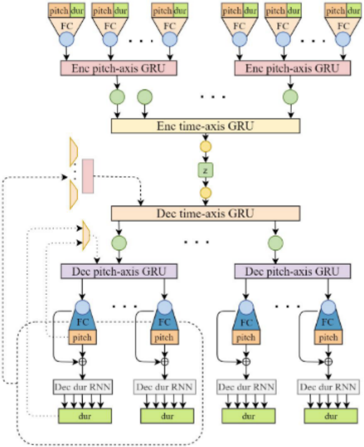


Figure: An overview of the model architecture. The recurrent layers are represented by rectangles and the fully-connected (FC) layers are represented by trapezoids. The *note*, *simu_note* and *score* events are represented by circles.

References I

[Rocca, 2019b] [Rocca, 2019a] [Learning and Simulation, 2021]
[Pati et al., 2019] [Chen et al., 2020] [Wang et al., 2020]
[Kingma and Welling, 2013] [Roberts et al., 2018]

Chen, K., Wang, C. I., Berg-Kirkpatrick, T., and Dubnov, S. (2020).
Music sketchnet: Controllable music generation via factorized representations of pitch and rhythm.
arXiv.

Kingma, D. P. and Welling, M. (2013).
Auto-encoding variational bayes.
arXiv preprint arXiv:1312.6114.

Learning, M. and Simulation (2021).
Variational Inference — Evidence Lower Bound (ELBO) — Intuition Visualization —.
https://www.youtube.com/watch?v=HxQ94L8n0vU.
[Online; accessed September-2021].

Pati, A., Lerch, A., and Hadjeres, G. (2019).
Learning to traverse latent spaces for musical score inpainting.
Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, pages 343–351.

References II

Roberts, A., Engel, J., Raffel, C., Hawthorne, C., and Eck, D. (2018).
A hierarchical latent vector model for learning long-term structure in music.
In *International Conference on Machine Learning*, pages 4364–4373. PMLR.

Rocca, J. (2019a).
Bayesian inference problem, MCMC and variational inference.
`https://towardsdatascience.com/bayesian-inference-problem-mcmc-and-variational-inference-25a8aa9`
[Online; accessed September-2021].

Rocca, J. (2019b).
Understanding Variational Autoencoders.
`https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73`.
[Online; accessed September-2021].

Wang, Z., Zhang, Y., Zhang, Y., Jiang, J., Yang, R., Zhao, J., and Xia, G. (2020).
Pianotree vae: Structured representation learning for polyphonic music.
arXiv preprint arXiv:2008.07118.