Deep Learning Techniques for Music Generation
Compound and GAN (6)

Jean-Pierre Briot

Jean-Pierre.Briot@lip6.fr

Laboratoire d'Informatique de Paris 6 (LIP6)
Sorbonne Université – CNRS

Programa de Pós-Graduação em Informática (PPGI)
UNIRIO

Architectures

Architectures

- *Feedforward* — *mini-bach.py*
- *Autoencoder* — *auto-bach.py*
  - *Variational Autoencoder (VAE)* — *VRAE*
- *Recurrent (RNN)*
  - *LSTM* — *lstm.py, Celtic*

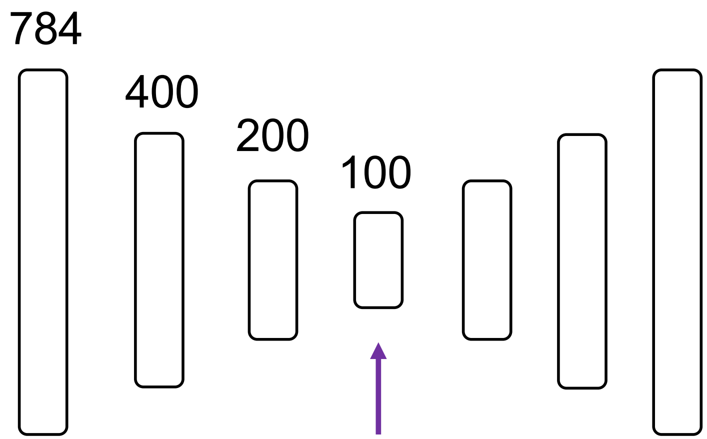
- Generative Adversarial Networks (GAN)

- Restricted Boltzmann Machine (RBM)

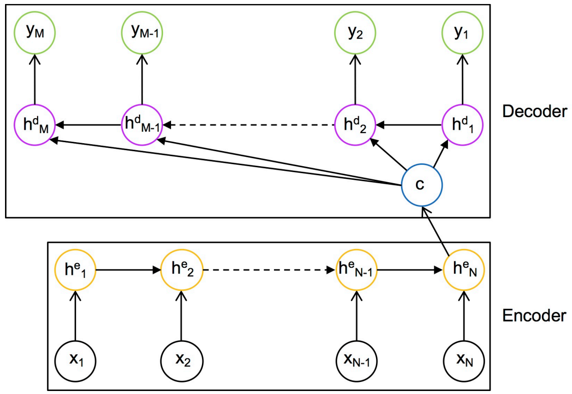
- Reinforcement Learning (RL)

Compound Architectures

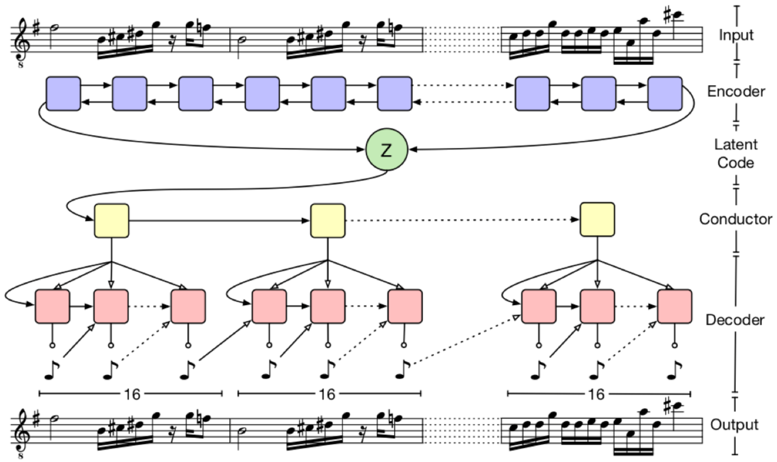
- Autoencoder Stack = Autoencoder^n
  - DeepHear, auto-bach.py



- Autoencoder(RNN, RNN) = RNN Encoder-Decoder
  - VRAE



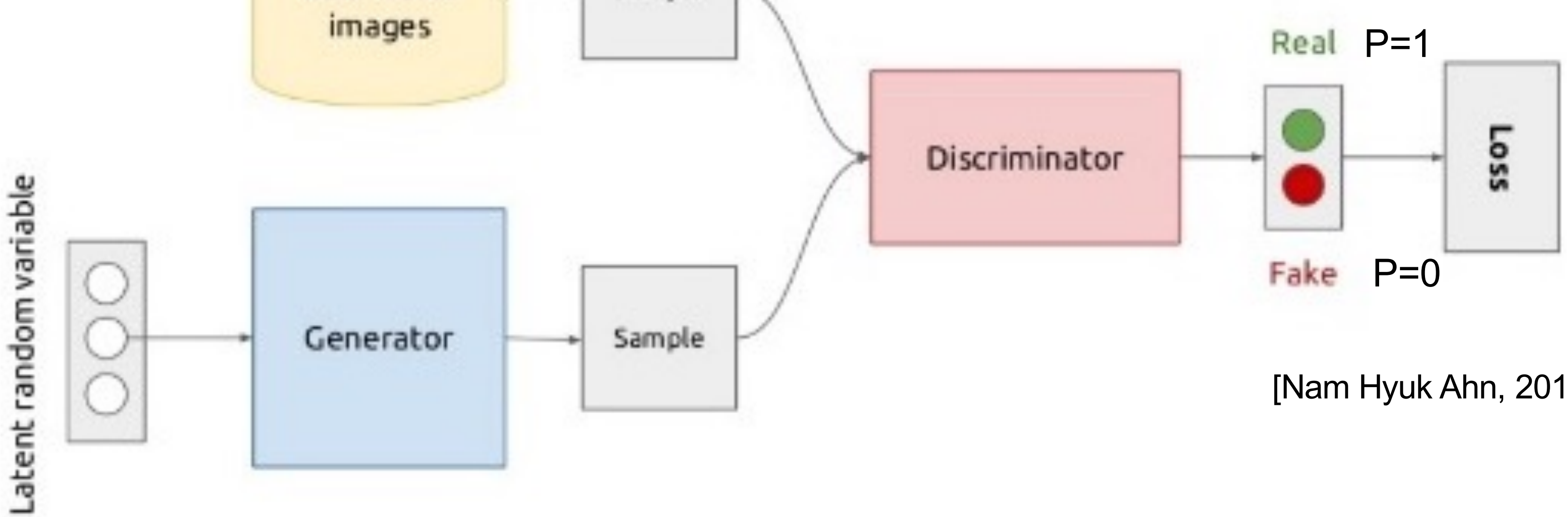
- RNN Variational Encoder-Decoder
  - Music-VAE



Generative Adversarial Networks (GAN) [Goodfellow et al., 2014]

- Training Simultaneously 2 Neural Networks
  - Generator
    - » Transforms Random noise Vectors into *Faked* Samples
  - Discriminator
    - » Estimates probability that the Sample came from training data rather than from G
  - Minimax 2-player game $\min_G \max_D V(G,D) = log(D(x)) + log(1 - D(G(z)))$

	Prediction by D
$D(x)$: P_D(x from real data)	*(Correct)*
$D(G(z))$: P_D(G(z) from real data)	*(Incorrect)*
1 - $D(G(z))$: P_D(G(z) from Generator)	*(Correct)*



[Nam Hyuk Ahn, 2017]

GAN Equation

- Binary Cross-Entropy:
- $H_B(y, \hat{y}) = -(y \log \hat{y} + (1-y) \log (1-\hat{y}))$

- $D(x) = 1$ $\quad$ P_D(x *from real data*) $\quad$ *Correct*

- $H_B(D(x), \hat{D}(x)) = -(D(x) \log \hat{D}(x) + (1-D(x)) \log (1-\hat{D}(x)))$
- $H_B(D(x), \hat{D}(x)) = -\log D(x)$

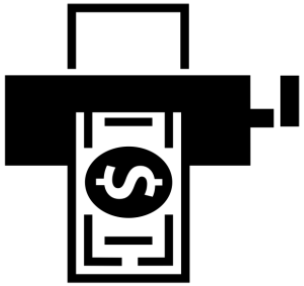
- $D(G(z)) = 0$ $\quad$ P_D(G(z) *from real data*) $\quad$ *Incorrect*

- $H_B(D(G(z)), \hat{D}(G(z))) = -(D(G(z)) \log \hat{D}(G(z)) + (1-D(G(z))) \log (1-\hat{D}(G(z))))$
- $H_B(D(G(z)), \hat{D}(G(z))) = -\log (1-\hat{D}(G(z)))$

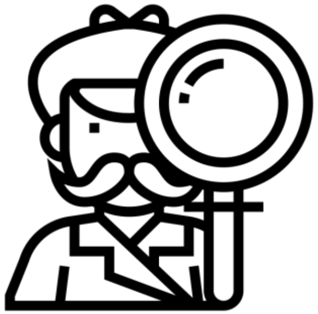
- $H_B(D(x), \hat{D}(x)) + H_B(D(G(z)), \hat{D}(G(z))) = -(\log \hat{D}(x) + \log (1-\hat{D}(G(z))))$

$$\min_G \max_D V(G,D) = log(D(x)) + log(1 - D(G(z)))$$

GAN and Turing Test

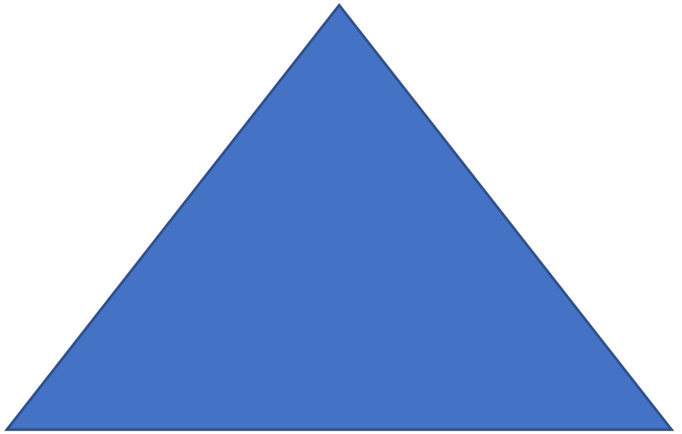


Generator



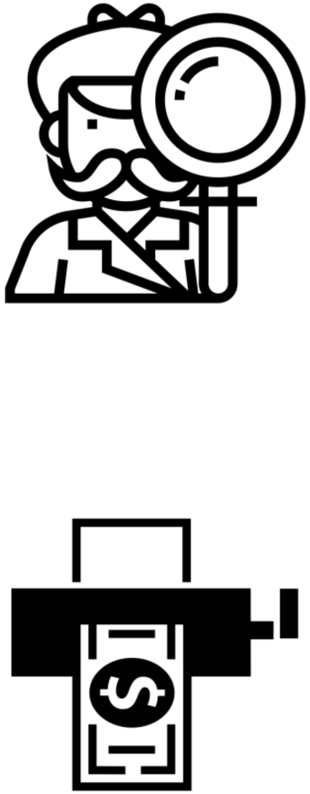
Discriminator

$D(G(z))$ or $D(x)$



$G(z)$ or x

z



[Goodfellow, 2016]

GAN Basic Training Algorithm

- Initialize $\theta^{(G)}, \theta^{(D)}$
- For $t = 1: b: T$
  - Initialize $\Delta\theta^{(D)} = 0$
  - For $i = t: t + b - 1$
    - Sample $z_i \sim p(z_i)$
    - Compute $D\big(G(z_i)\big), D(x_i)$
    - $\Delta\theta_i^{(D)} \leftarrow$ Compute gradient of **Discriminator loss**, $J^{(D)}\big(\theta^{(G)}, \theta^{(D)}\big)$
    - $\Delta\theta^{(D)} \leftarrow \Delta\theta^{(D)} + \Delta\theta_i^{(D)}$
  - Update $\theta^{(D)}$
  - Initialize $\Delta\theta^{(G)} = 0$
  - For $j = t: t + b - 1$
    - Sample $z_j \sim p(z_j)$
    - Compute $D\left(G(z_j)\right), D(x_j)$
    - $\Delta\theta_j^{(G)} \leftarrow$ Compute gradient of **Generator loss,** $J^{(G)}\big(\theta^{(G)}, \theta^{(D)}\big)$
    - $\Delta\theta^{(G)} \leftarrow \Delta\theta^{(G)} + \Delta\theta_j^{(G)}$
  - Update $\theta^{(G)}$

Examples of GAN Generated Images

CelebFaces Attributes Dataset (CelebA)
> 200K celebrity images

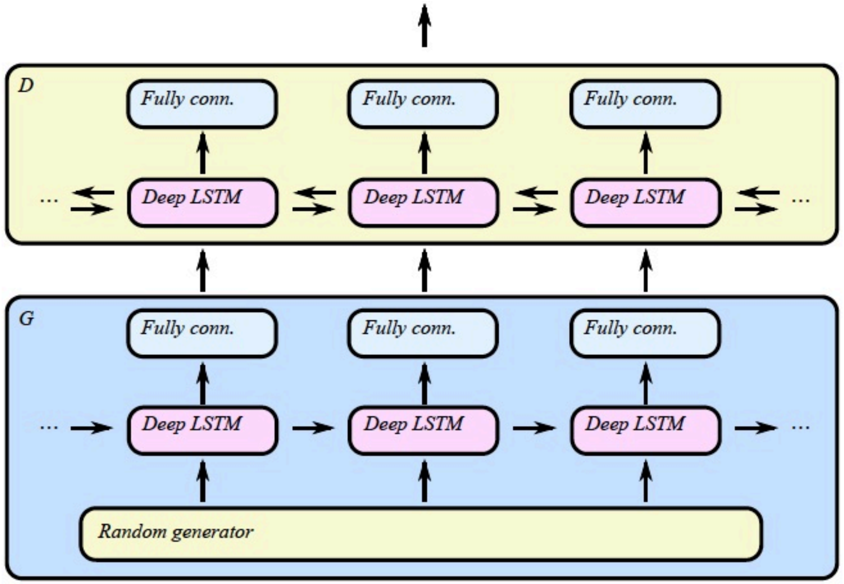
[Brundage et al., 2018]

Synthetic (Generated) Celebrity images

[Karras et al., 2018]

C-RNN-GAN [Mogren, 2016]

GAN(Bidirectional-LSTM², LSTM²)



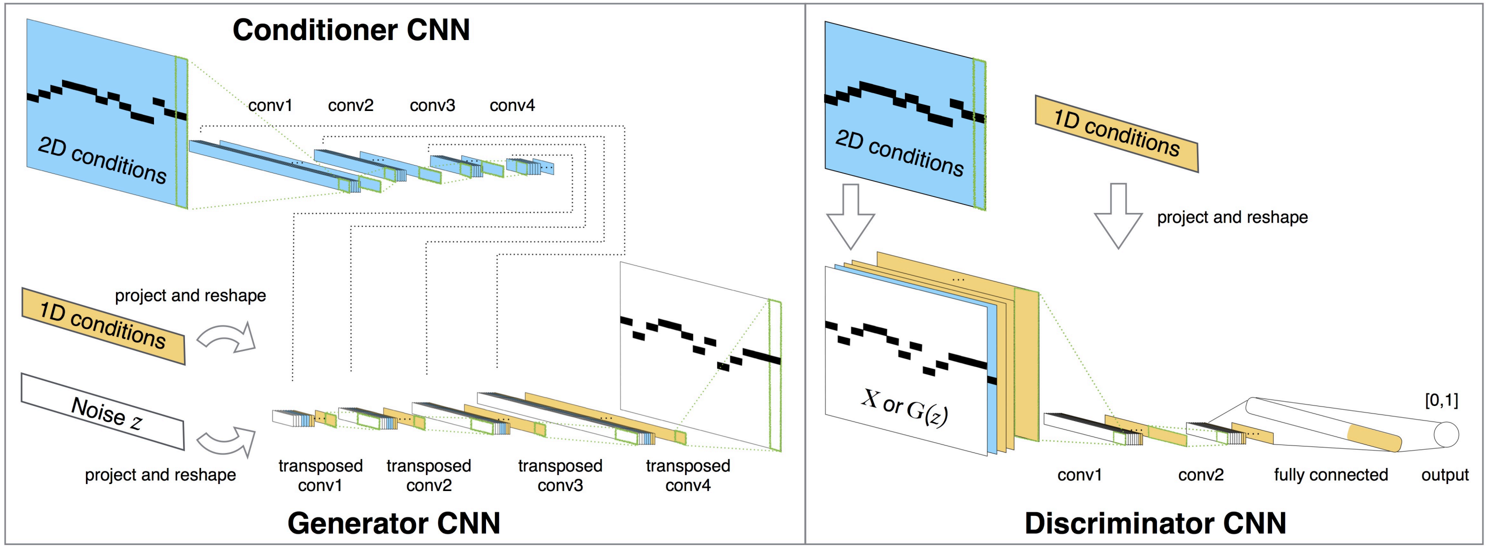
- Discriminator considers the hidden layers (forward and backward) values to be (or not) representative of the Real data
  - Analog to RNN Encoder-Decoder which considers the hidden layer as the summary of a sequence

- Classical music Training Dataset

MidiNet [Yang et al., 2017]

GAN(Conditioning(Convolutional(Feedforward),
Convolutional(Feedforward(History, Chord sequence))),
Conditioning(Convolutional(Feedforward), History))

- Convolutional
- Conditioning
  - Previous measure
  - Chord sequence



- Pop music Training Dataset

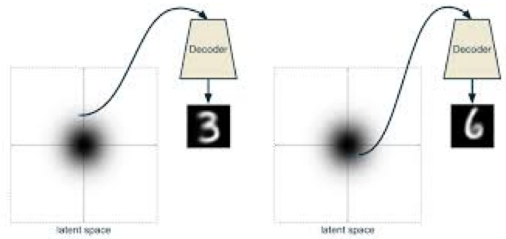
https://soundcloud.com/vgtsv6jf5fwq/model3

VAE vs GAN

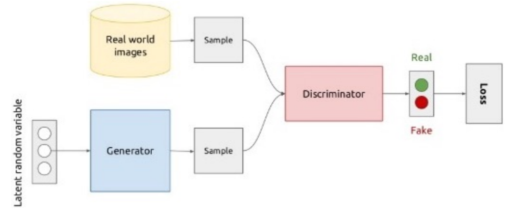
- VAE (Variational Autoencoder) and GAN (Generative Adversarial Networks)

Some Similarities:

- Are both generative architectures
- Generate from random latent variables



[Dykeman, 2016]

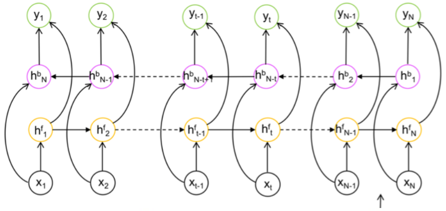


Differences:

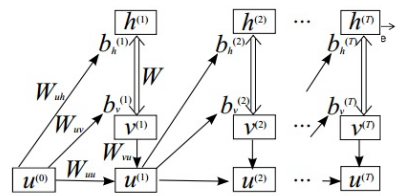
- VAE is representational of the whole training dataset
- GAN is not
- Smooth control interface for exploring latent data space
- GAN has (ex: interpolation) but not as for VAE
- GAN produces better quality content (ex: better resolution images)

Compound Architectures

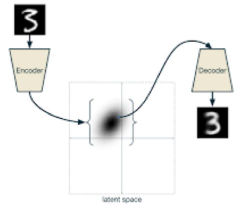
- Composition
  - Bidirectional RNN, combining two RNNs, forward and backward in time



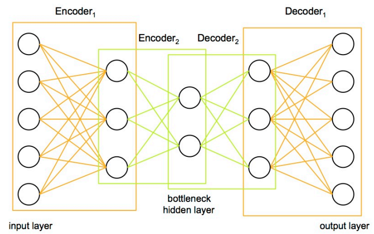
  - RNN-RBM [Boulanger-Lewandowski et al., 2012], combining an RNN (horizontal/sequence) and an RBM (vertical/chords)

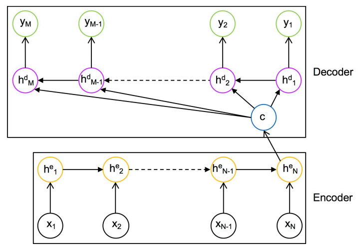


- Refinement
  - Sparse autoencoder
  - Variational autoencoder (VAE) = Variational(Autoencoder)



- Nested
  - Stacked autoencoder = Autoencoder^n
  - RNN Encoder-Decoder = Autoencoder(RNN, RNN)





- Pattern instantiation
  - C-RBM [Lattner et al., 2016] = Convolutional(RBM)
  - C-RNN-GAN [Mogren, 2016] = GAN(Bidirectional-LSTM^2, LSTM^2)
  - Anticipation-RNN [Hadjeres & Nielsen, 2017] = Conditioning(RNN, RNN)