Symbolic Music Genre Transfer with CycleGAN

MUS 206

Outline

- Task
- CycleGAN
- CycleGAN for our task
- Dataset
- Input-Output
- Training
- Results
- Contributions

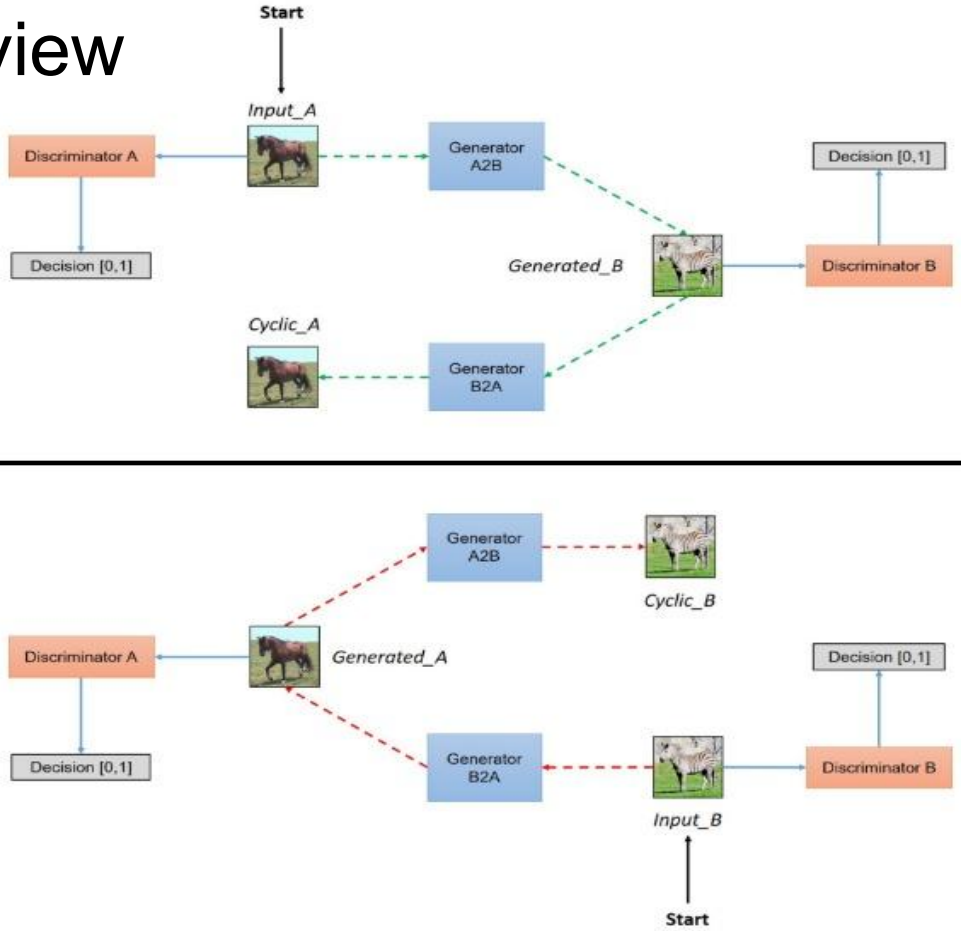
Musical Task - Symbolic Music Genre Transfer with CycleGAN

- Deep generative models such as Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) have recently been applied to style and domain transfer for images, and in the case of VAEs, music.
- GAN-based models employing several generators and some form of cycle consistency loss have been among the most successful for domain transfer.
- In this project we apply such a model to symbolic music and show the feasibility of our approach for music genre transfer.
- Evaluations using separate genre classifiers show that the style transfer can work.

Method that we will be trying out:

- CycleGAN (Cycle Generative Adversarial Network)
  - Closely follow the paper “Symbolic Music Genre Transfer with CycleGAN”
  - In order to improve the fidelity of the transformed music, we add additional discriminators that cause the generators to keep the structure of the original music mostly intact, while still achieving strong genre transfer.
  - After replicating the paper, and it’s results we will run the model on custom datasets.

Cycle GANs Overview



CycleGAN for this task

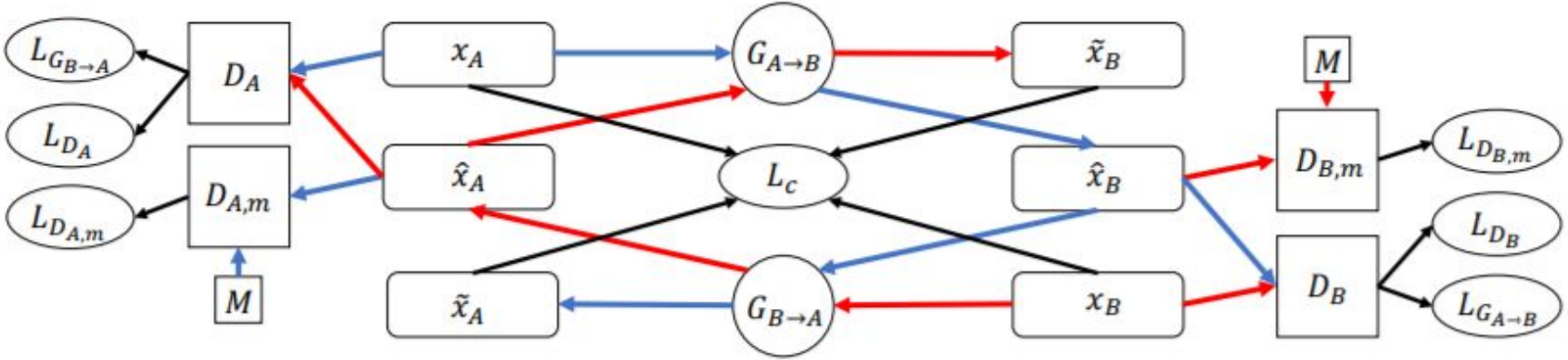


Fig. 1. Architecture of our model. The two cycles are shown in blue and red respectively. The black arrows point to the loss functions. We extend the basic CycleGAN architecture with additional discriminators $D_{A,m}$ and $D_{B,m}$.

Please note that a detailed explanation of the image is given on the next page. Image source : https://www.tik.ee.ethz.ch/file/2e6c8407bf92ce1e47c0faa7e9a3014d/cyclegan-music-style%20(3).pdf

Explaining the Cycle GAN Architecture

- $G_{A \to B}$ and $G_{B \to A}$ are two generators which transfer data between A and B.
- D_A and D_B are two discriminators which distinguish if data is real or fake.
- $D_{A,m}$ and $D_{B,m}$ are two extra discriminators which force the generators to learn more high-level features.
- Following the blue arrows, x_A denotes a real data sample from source domain A.
- $\hat{x}_B$ denotes the same data sample after being transferred to target domain B, i.e., $\hat{x}B = G_{A \to B}(x_A)$
- $\tilde{x}_A$ denotes the same data sample after being transferred back to the source domain A, i.e., $\tilde{x}_A = G_{B \to A}(G_{A \to B}(x_A))$. Equivalently, following the red arrows describes the opposite direction, i.e., the transfer from B to A and back to B.
- M is a dataset containing music from multiple domains, e.g, $M = A \cup B$. xM denotes a data sample from M

Loss Function

$$L_{G_{A\to B}} = \|D_B(\hat{x}_B) - 1\|_2$$

$$L_{G_{B\to A}} = \|D_A(\hat{x}_A) - 1\|_2$$

To enforce forward-backward consistency, Zhu et al. [2] introduce an extra L1 loss term called *cycle consistency loss*:

$$L_c = \|\bar{x}_A - x_A\|_1 + \|\bar{x}_B - x_B\|_1$$

$$L_G = L_{G_{A\to B}} + L_{G_{B\to A}} + \lambda L_c$$

$$L_{D_A} = \frac{1}{2}\left(\|D_A(x_A) - 1\|_2 + \|D_A(\hat{x}_A)\|_2\right)$$

$$L_{D_B} = \frac{1}{2}\left(\|D_B(x_B) - 1\|_2 + \|D_B(\hat{x}_B)\|_2\right)$$

$$L_{D_{A,m}} = \frac{1}{2}\left(\|D_{A,m}(x_M) - 1\|_2 + \|D_{A,m}(\hat{x}_A)\|_2\right)$$

$$L_{D_{B,m}} = \frac{1}{2}\left(\|D_{B,m}(x_M) - 1\|_2 + \|D_{B,m}(\hat{x}_B)\|_2\right)$$

where M denotes mixed real data from multiple domains (here possibly Jazz, Classic and/or Pop). Thus the total loss for the discriminators is

$$L_{D,all} = L_D + \gamma\left(L_{D_{A,m}} + L_{D_{B,m}}\right) \quad (2)$$

Dataset

- Our Dataset will be comprised of three genres of songs:
  - [Jazz - *around 12k*, Classical - *around 16k*, Pop - *around 20k*]
- Pre-Processing:
  - Using pypianoroll and pretty midi
  - Resample midi file for time discretization and obtaining a matrix representation
  - *t* represents number of time steps = 16 per bar (common sampling rate choice in the current literature)
  - Ignore note loudness (all notes sound the same)
  - Retain as many notes as possible from multiple tracks : merge the notes from all the tracks
  - *p* represents number of possible pitches = 84 (Notes below C1 and above C8 are ignored)
  - the final input representation comes out to be 64 x 84 for a 4 bar piece

Input-Output

- Input would be a piano roll representation of shape *t x p*
- The output would be the same representation of shape *t x p*
  - *Cannot obtain the original MIDI back*
  - Convert the *t x p* representation back to a midi file

Important Training Details

- Batch Size : 32
- Gaussian Noise at Discriminator
  - sigma = 0 for partial model
  - sigma = 1 for base model
- Adam Optimizer
  - alpha = 0.0002
  - momentum decay : 0.5 & 0.99
- For cycle consistency loss
  - lambda = 10
- Weightage of the two “extra” D
  - gamma = 1
- Epochs
  - For around 20 epochs

TABLE I
DISCRIMINATOR ARCHITECTURE

Input: $(batchsize \times 64 \times 84 \times 1)$					
layer	filter	stride	channel	instance norm	activation
conv	4×4	2×2	64	False	LReLu
conv	4×4	2×2	256	True	LReLu
conv	1×1	1×1	1	False	None
Output: $(batchsize \times 16 \times 21 \times 1)$					

TABLE II
GENERATOR ARCHITECTURE

Input: $(batchsize \times 64 \times 84 \times 1)$					
layer	filter	stride	channel	instance norm	activation
conv	7×7	1×1	64	True	ReLu
conv	3×3	2×2	128	True	ReLu
conv	3×3	2×2	256	True	ReLu
$10\times$ ResNet	3×3	1×1	256	True	ReLu
	3×3	1×1	256	True	ReLu
deconv	3×3	2×2	128	True	ReLu
deconv	3×3	2×2	64	True	ReLu
deconv	7×7	1×1	1	False	Sigmoid
Output: $(batchsize \times 64 \times 84 \times 1)$					

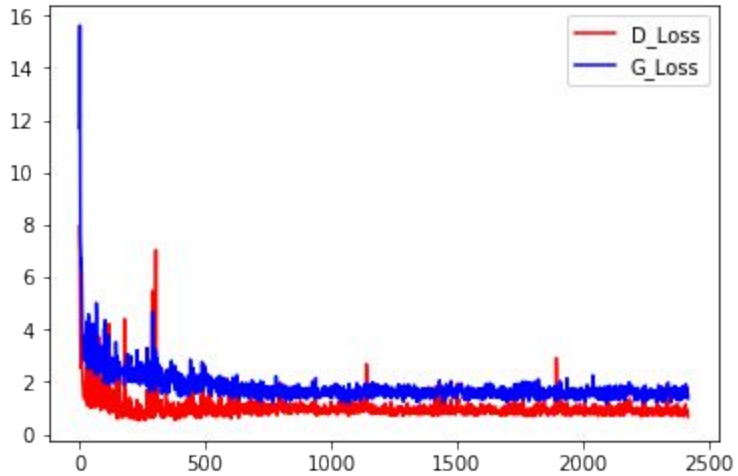
Experimentations

- Between different types of noise
  - Gaussian Noise
  - B&W Noise
  - Uniform Noise
- Experimenting with $D_{A,m}$ and $D_{B,m}$
  - What should M comprise of?
    - A & B
    - A & B & C

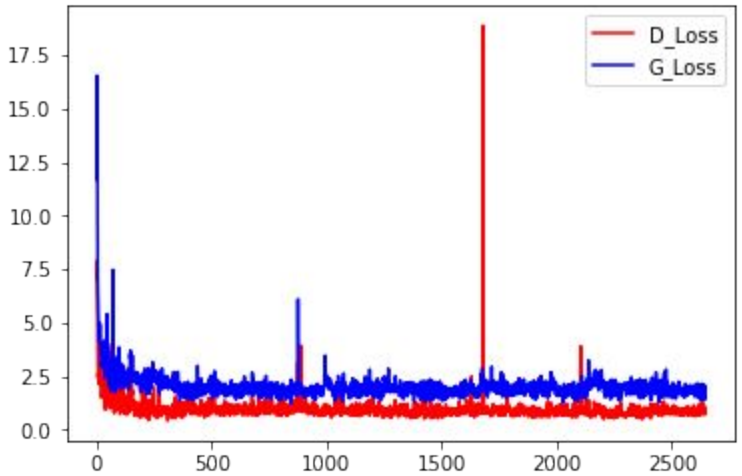
Best Performing Model Details :

- Gaussian Noise
- All types of genres used in $D_{A,m}$ and $D_{B,m}$

Training Loss Graphs: Pop - Classical



Training Loss Graphs : Jazz - Classical



Results

Listen to some of our results!

Future Work

- Don’t merge tracks (guitar, piano etc), include them as different channels?
- Try a different architecture altogether maybe?
- Try using LSTM with Reward Function for genre style transfer instead on Conv-Nets

References

1. Symbolic Music Genre Transfer with CycleGAN, Brunner et al.
2. DeepJazz, https://github.com/jisungk/deepjazz.
3. CycleGAN, https://hardikbansal.github.io/CycleGANBlog/
4. CycleGAN, https://junyanz.github.io/CycleGAN/
5. MIDINet, https://arxiv.org/abs/1703.10847.
6. Neural Translation of Musical Style, imanmalik.com/cs/2017/06/05/neural-style.html.
7. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient, https://arxiv.org/pdf/1609.05473.pdf