Project #4

Polyphonic Piano Melody Generation

1. 한창진/산업공학과/2019-170
2. 김 태/기계공학과/2019-321

Contents

1. Introduction

본 프로젝트에서는 16 마디의 polyphonic melody 를 가지는 midi 파일을 data set 으로, GAN 모델을 사용, 학습하여 적절한 샘플 곡을 생성하도록 하는 것이 목적이다. 이를 위해 마디 당 16 분 음표 16 개로 구성된 2,561 곡의 midi 파일이 주어졌다. GAN 모델의 학습 시 기존의 다른 프로젝트와는 달리 모델의 학습 적합도록 정량적으로 판단할 수 있는 기준이 없어, 생성된 곡을 매 epoch 마다 저장하여 학습이 잘 이루어졌다고 판단된 epoch 에서의 곡을 샘플로 제출하였다.

.

2. Literature Review

1) Generative Adversarial Network (GAN)[1][2]

GAN 은 생성자(generator)와 구분자(discriminator) 두 네트워크를 적대적으로 학습시키는 비지도 학습 기반의 생성모델이다. Generator 는 noise 를 받아 실제 데이터와 비슷한 데이터를 만들어내도록 학습되며, discriminator 는 generator 가 생성한 가짜 데이터를 구별하도록 학습된다. 이 모델의 궁극적 목적은 실제 데이터의 분포에 가까운 데이터를 생성하는 것이라 이해할 수 있다. GAN 의 목적함수는 아래와 같이 표현할 수 있으며, generator 와 discriminator 가 경쟁하며 균형점을 찾아가는 방식이라 할 수 있다.

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log\{1 - D(G(z))\}]$$

GAN 은 발표 이후 다양한 분야에 활용되어 왔으나, 학습이 불안정하여 다양한 분야에 응용되는 것에 제약이 있었다.

2) Deep Convolutional GAN (DCGAN)[2][3]

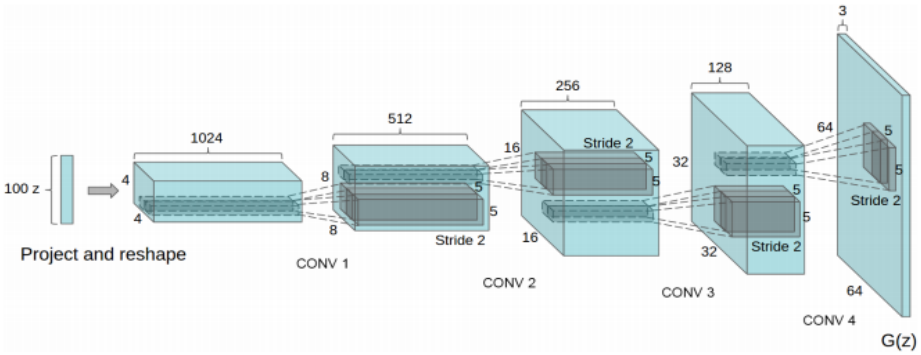

Figure 1. DCGAN generator

DCGAN 은 위에서 언급한 GAN 을 개선한 모델로, 안정적인 학습이 가능하도록 하였다. DCGAN 의 특징으로는 먼저 선형 layer 와 pooling layer 를 최대한 배제하고 convolution 과 Transposed Convolution 으로 네트워크 구조를 만들었다. 또한 batch normalization 을 사용하여 layer 의 입력 데이터가 분포가 치우쳐져 있을 때 mean 과 variance 를 조정해주는 역할을 한다.

또한 마지막 layer 를 제외하고 생성자의 모든 layer 에 ReLU 를, 구분자의 모든 layer 에 Leaky-ReLU 를 사용했다. 이와 더불어 가장 좋은 optimization algorithm 를 사용하고 적절한 learning rate 을 적용한 부분은 다양한 조건에서 직접 비교하여야 얻을 수 있는 부분들을 찾아낸 것으로 보여진다. 이러한 DCGAN 은 GAN 모델이 보다 널리 활용되는 데에 결정적인 역할을 했다.

구분	Generator	Discriminator
Pooling Layers	Not Used. But use strided convolutions instead.	same
Batch Normalization	Use except output layer	Use except input layer
Fully connected hidden layers	Not used	Not used
Activation function	ReLU for all layers except for the output, which uses Tanh	Leaky-ReLU for all layers

Table 1 Characteristics of DCGAN

3. Model Structure

본 프로젝트에서 구성한 Model 의 구조는 아래와 같다.

Discriminator 는 총 4 개의 convolution layer 를 쌓고 이를 Time distributed layer 와 dense layer 를 차례로 통과시켜 decision 을 출력하도록 학습시켰고, Generator 는 input noise 를 받아들이는 1 개의 dense layer 와 총 5 개의 deconvolution layer 를 쌓아서 sample 을 generate 할 수 있도록 학습시켰다. 그리고 학습을 안정적으로 진행하기 위해 Batch normalization 과 activation, dropout 을 적용하였는데 각각이 적용된 시점은 아래 그림에 묘사되어 있다.

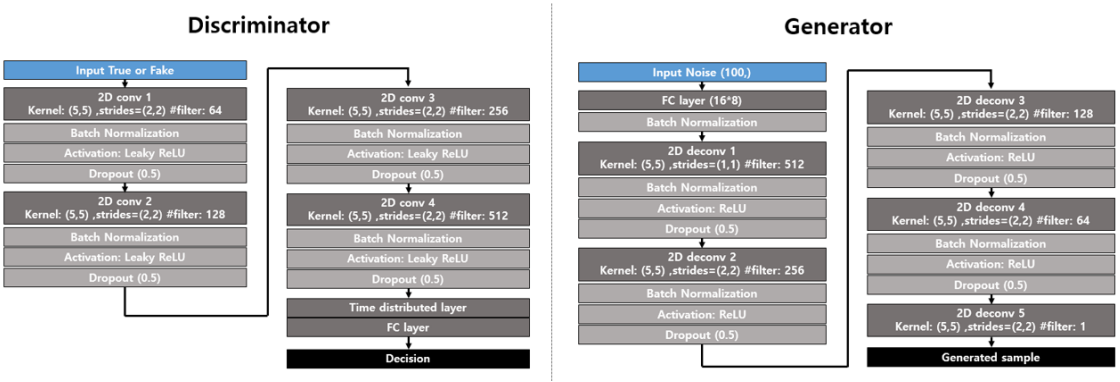

Figure 2. Model Sturcture for this project (Discriminator and Generator)

4. Result & Conclusion

Generator loss 와 Discriminator loss 그리고 Gradient penalty loss 까지 세 loss 가 모두 작을 때 모델이 실제와 유사한 음악을 생성하고 있는 것이라 판단하여 이 때의 모델 가중치 값들을 저장하고 샘플링을 진행하기로 하였다. 아래의 loss 변화 추이로 보아 총 500 epoch 을

돌려보았을 때, 100~200 epoch 사이의 결과들이 적합한 모델 후보로 거론되었는데 샘플링 된 음악을 들으며 테스트 해본 결과 160 epoch 에서 성능이 가장 좋다고 판단했다.

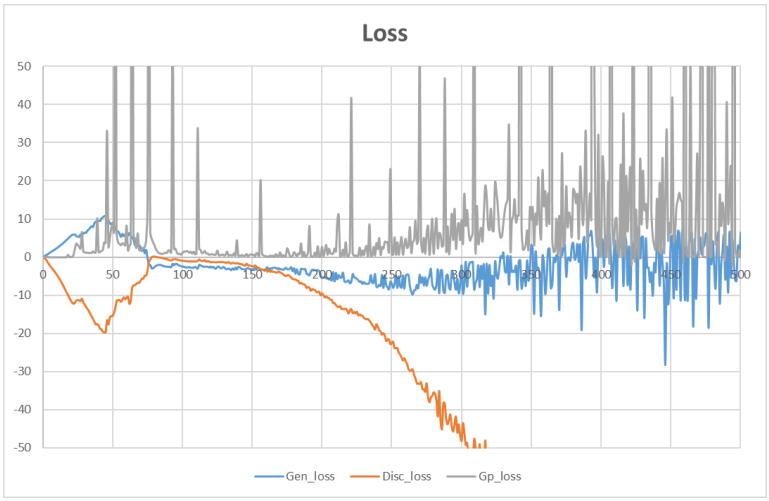

Figure 3. Loss during learning epoch

아래의 그림은 1~190 epoch 동안 학습이 진행됨에 따라 샘플링 된 미디 파일을 piano roll 로 표현했을 때 변화하는 양상을 보여준다. 학습을 하면서 많은 음을 동시에 연주하는 경향이 없어지고 음의 높낮이가 변화하는 것을 관찰할 수 있으므로 의미있는 학습이 이루어지고 있다고 볼 수 있다.

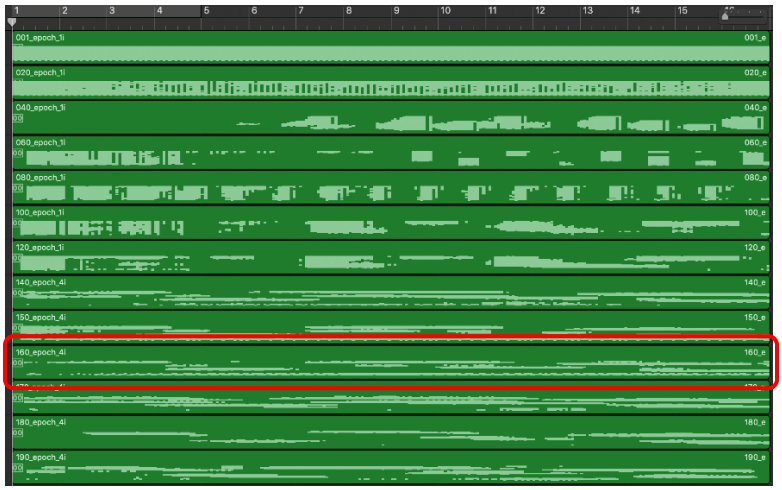

Figure 4. Piano rolls for sampled midi

* Reference

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. “Generative adversarial nets,” Proc. Advances in Neural Information Processing Systems, pages 2672–2680, 2014.

[2] https://ratsgo.github.io/generative%20model/2017/12/21/gans/

[3] Alec Radford & Luke Metz, Soumith Chintala, “UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS,” ICLR, 2016.