

MyMAI_Music Translation



딥러닝을 활용한 Singing Voice Conversion

김유민

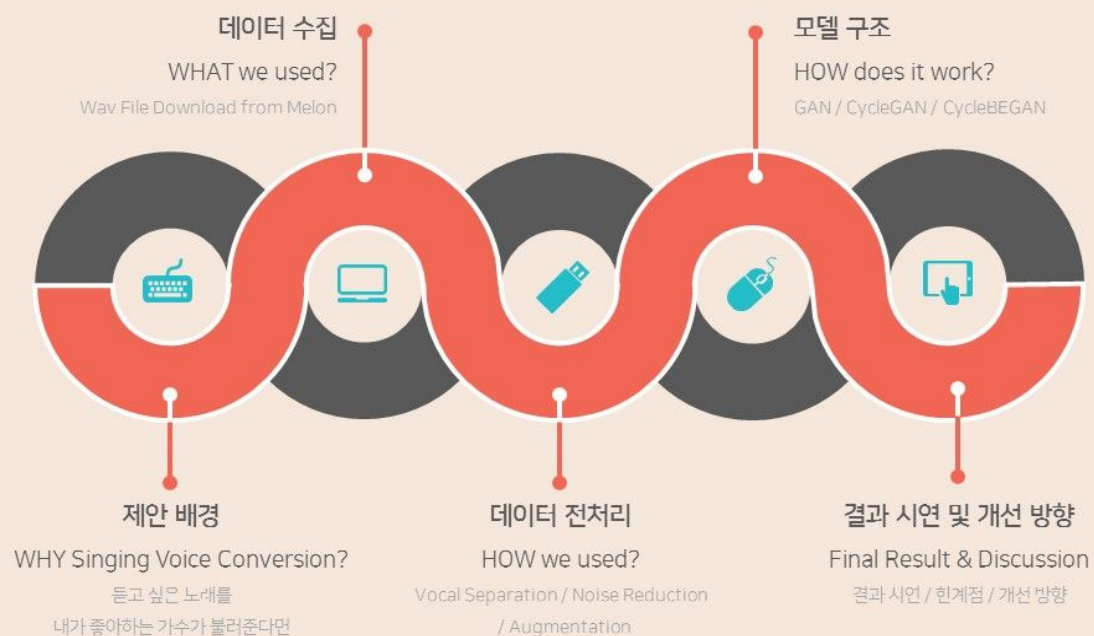
박진혁

이소라

정혜인

조민호

최승호



01. Introduction

WHY Singing Voice Conversion?

Tacotron을 이용한 Voice Conversion

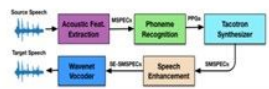
TACO-VC: A SINGLE SPEAKER TACOTRON BASED VOICE CONVERSION WITH LIMITED DATA

Roe Levy-Leshem, Raja Giryes

School of Electrical Engineering, Tel Aviv University, Tel Aviv,

ABSTRACT

This paper introduces Taco-VC, a novel architecture for voice conversion (VC) based on the Tacotron synthesizer, which is a sequence-to-sequence with attention model. The training of multi-speaker voice conversion systems requires a large amount of



책 읽어주는 딥러닝 - 유인나 Ver.

제7회 투빅스 컨퍼런스 - 투빅스랩소디



박승은 백광제 신현정 임진혁 전민규 정용호

TOBIG'S Rhapsody

tacotron을 이용한 음성 합성기 제작

다양한 "Speech" Voice Conversion 시도들!

"Singing" Voice Conversion은.....?

Q. 내가 듣고 싶은 노래를
내가 좋아하는 가수가 불러준다면..?





거미 - You Are My Everything

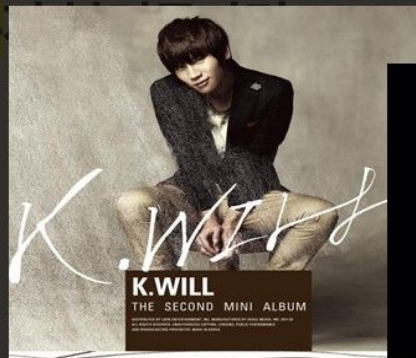
with 아이유's voice

내가 듣고 싶은 노래를

내가 좋아하는 가수

케이윌 - 내 생애 아름다운

with 10cm's voice



02. DATA

WHAT we used?

Q. 우리는 어떤 데이터가 필요할까?

A. “목소리만 남은” 노래 데이터가 필요하다!



-Original 가수-



-Conversion 가수-

각각 10곡 이상의 원곡과 Inst.파일 다운로드

ex) 아이유, 백지영, 거미, 산용재, 10cm, 케이윌 등



① 원곡 파일 & Inst.파일
각각 푸리에 변환



② 원곡과 Inst. 푸리에 변환값의
차이를 이용해 목소리 분리



③ 목소리 분리 값 to .wav 파일
by 역푸리에 변환

cf. What is Fourier Transform?

시간에 대한 신호를
함수를 구성하고 있는 주파수 성분으로 분해하는 작업

03. Preprocessing

HOW we used?

1. Preprocessing



[목소리 분리 후 .wav 파일]



1) 무음값 제거

의미 있는 Feature만 뽑을 수 있도록 필요 없는 부분 제거
ex. 가사가 없는 부분, 간주 부분, 숨쉬는 부분 등

1. Preprocessing



[목소리 분리 후 .wav 파일]



1) 무음값 제거

의미 있는 Feature만 뽑을 수 있도록 필요 없는 부분 제거
ex. 가사가 없는 부분, 간주 부분, 숨쉬는 부분 등



2) Notch Filter 적용

- 주파수 상에서 잡음으로 인식되는 Noise 제거
- 극도의 저주파와 고주파는 잡음과 동일함
→ 특정 범위를 벗어난 주파수는 잡음으로 인식

1. Preprocessing



[목소리 분리 후 .wav 파일]

1) 무음값 제거

의미 있는 Feature만 뽑을 수 있도록 필요 없는 부분 제거
ex. 가사가 없는 부분, 간주 부분, 숨쉬는 부분 등

2) Notch Filter 적용

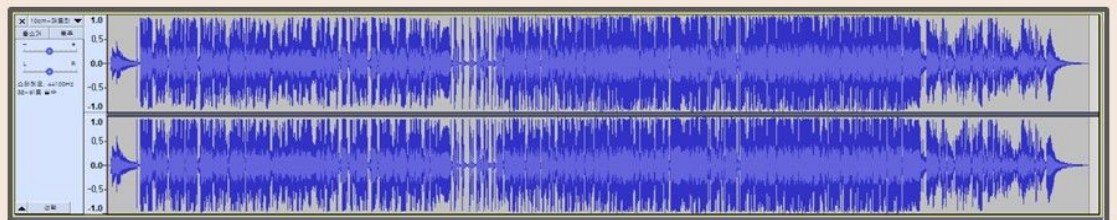
- 주파수 상에서 잡음으로 인식되는 Noise 제거
- 극도의 저주파와 고주파는 잡음과 동일함
→ 특정 범위를 벗어난 주파수는 잡음으로 인식

3) Noise Reduction

보다 세밀하게 곡 내에서의 잡음 인식 후 제거



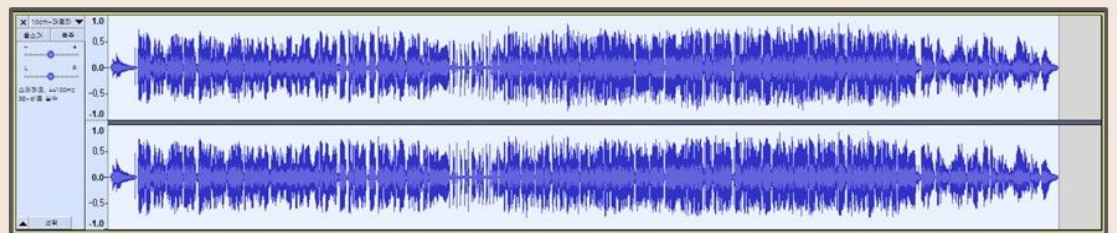
케이윌 - 내 생애 아름다운
(원곡)



Before



케이윌 - 내 생애 아름다운
(전처리 후)

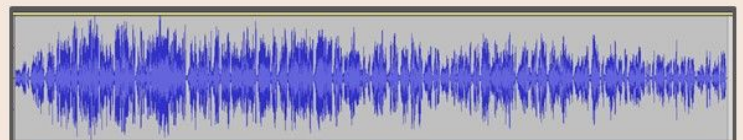


After

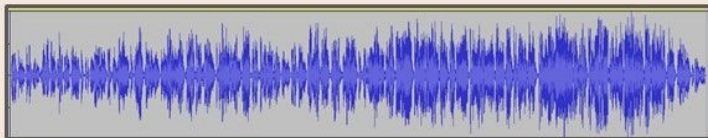
2. Augmentation



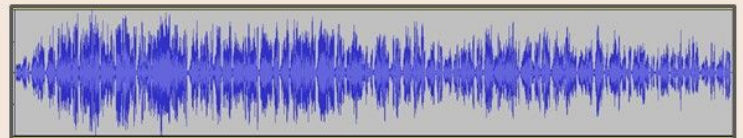
[원본]



[거꾸로 재생]



[위아래 뒤집기]



[거꾸로 재생 + 위아래 뒤집기]

2. Augmentation



[원본]

Augmentation을 통해
데이터 크기 4배 증강!



[거꾸로 재생]

White Noise, Stretch 등의 시도는
음정이 담긴 데이터라는 점에서 저하된 성능을 보임!



[위아래 뒤집기]



[거꾸로 재생 + 위아래 뒤집기]

04. MODEL

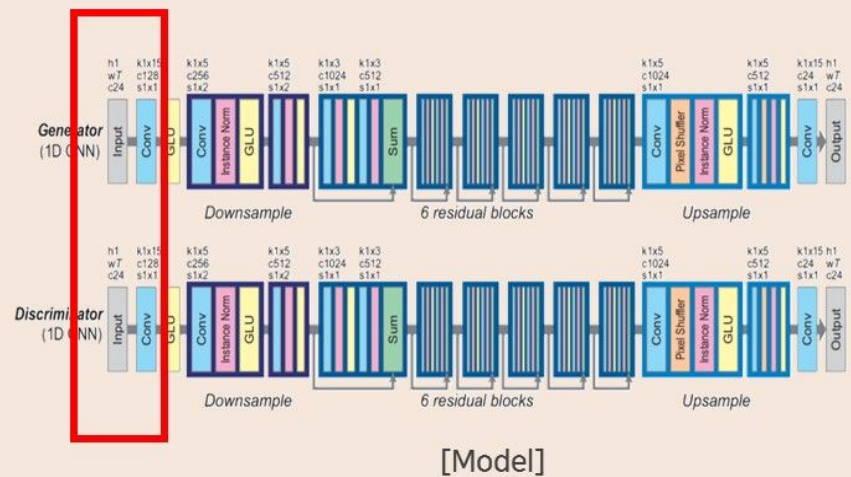
HOW we used?

Model Input



[.wav 파일]

Feature Extraction
: MCEP



Audio Analysis



[Raw Wave]



① F0

- 근음(뿌리근 소리 음)
- 소리를 들었을 때 사람이 인지하는 음의 뿌리가 되는 주파수



② Spectral Envelope

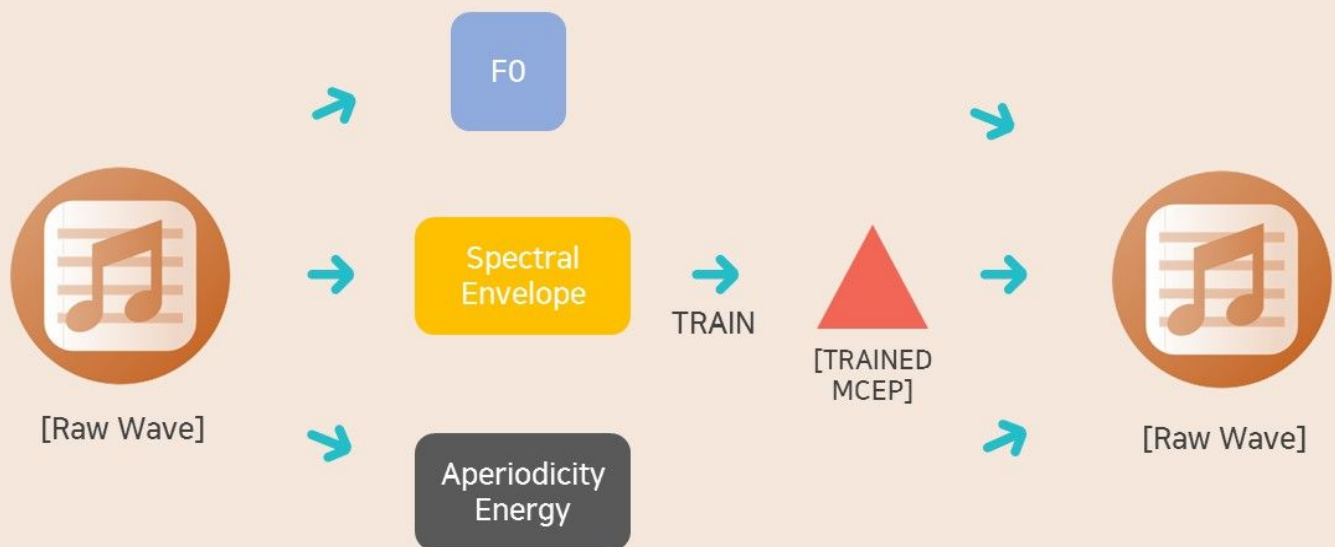
- Spectrum을 역푸리에 변환한 Cepstrum 중 저주파 부분



③ Aperiodicity Energy

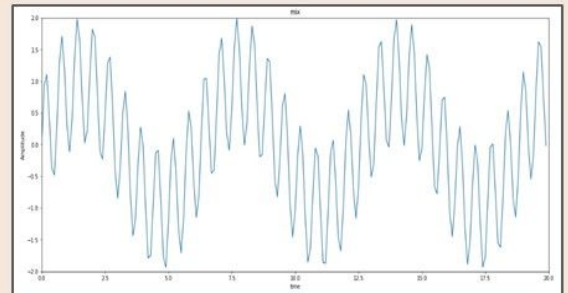
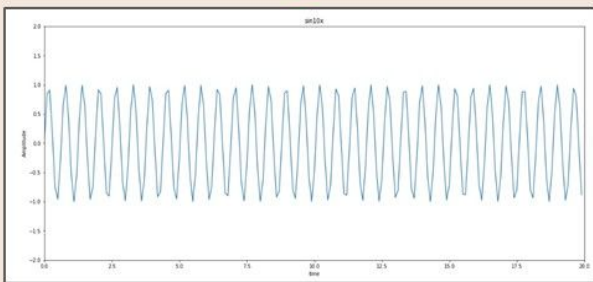
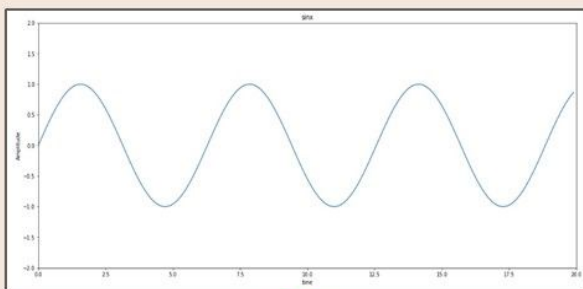
- 주파수의 극소값과 극대값을 연결한 curve 간의 ratio
- Envelope 이외 진동하는 진폭의 정보

Audio Analysis



04 모델 구조 - Feature Extraction

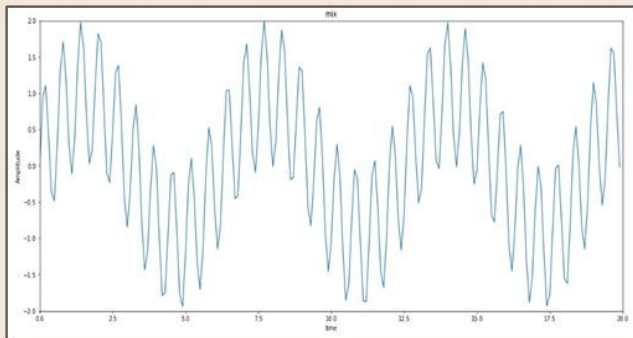
Fourier Transform



X축: Time(시간)
Y축: Amplitude(진폭)

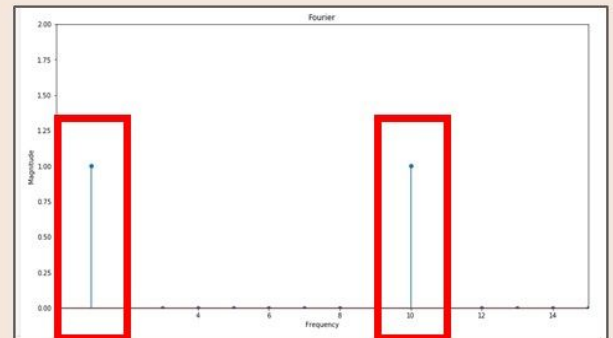
04 모델 구조 - Feature Extraction

Fourier Transform



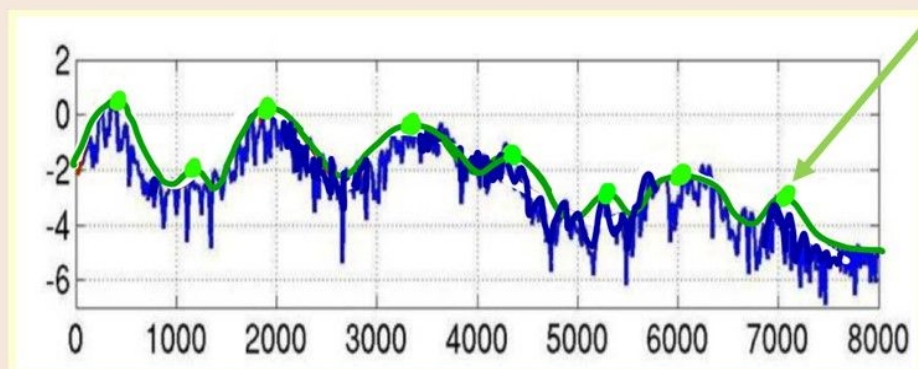
X축: Time(시간)
Y축: Amplitude(진폭)

FFT



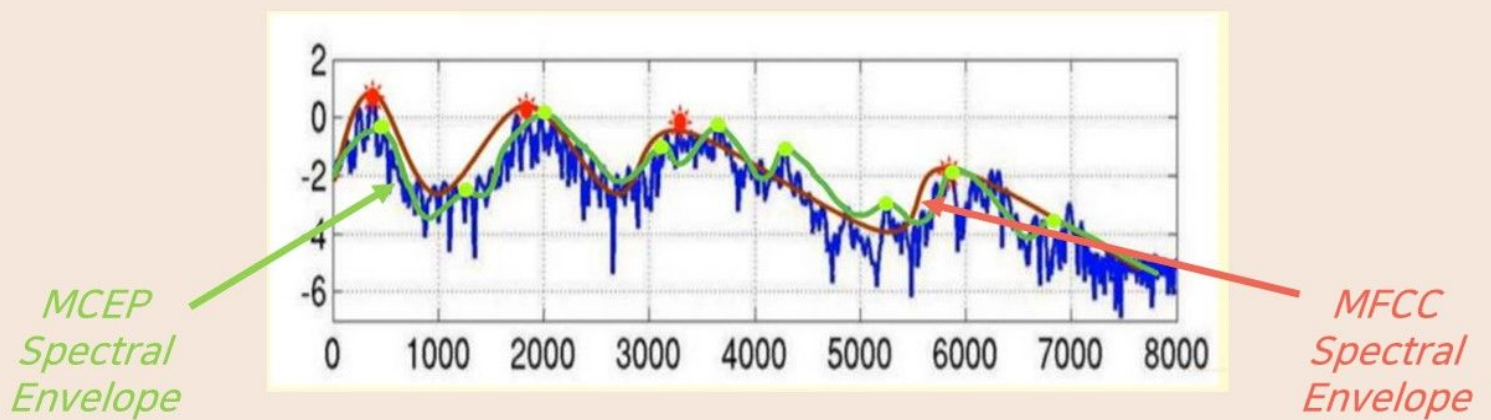
X축: Frequency(주파수)
Y축: Magnitude(주파수의 세기)

Q. What is Spectral Envelope?



Q. What is Spectral Envelope?

● MCEP VS ▲ MFCC



0. GAN(Generative Adversarial Network) - Background

<pix2pix>



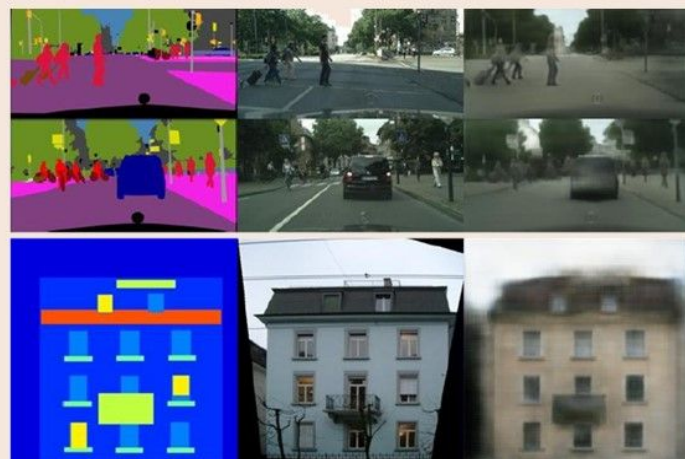
Input



Output



Ground Truth



Input

Ground Truth

L1

0. GAN(Generative Adversarial Network)

: Generator와 Discriminator 사이의 경쟁적 학습을 이용한 Network



0. GAN(Generative Adversarial Network) - Loss Function

: Generator와 Discriminator 사이의 경쟁적 학습을 이용한 Network

[Discriminator]

$$\min_G \max_D V(D, G) = \underbrace{\mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})]}_{\text{진짜 데이터 } \mathbf{x} \text{ 를 넣었을 때의 값}} + \underbrace{\mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))]}_{\text{G가 만든 가짜 데이터 } G(\mathbf{z}) \text{ 를 넣었을 때의 값}}$$

[Generator]

$$\min_G \max_D V(D, G) = \cancel{\mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})]} + \mathbb{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))]$$

0. GAN(Generative Adversarial Network) - Loss Function

: Generator와 Discriminator 사이의 경쟁적 학습을 이용한 Network

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))]$$

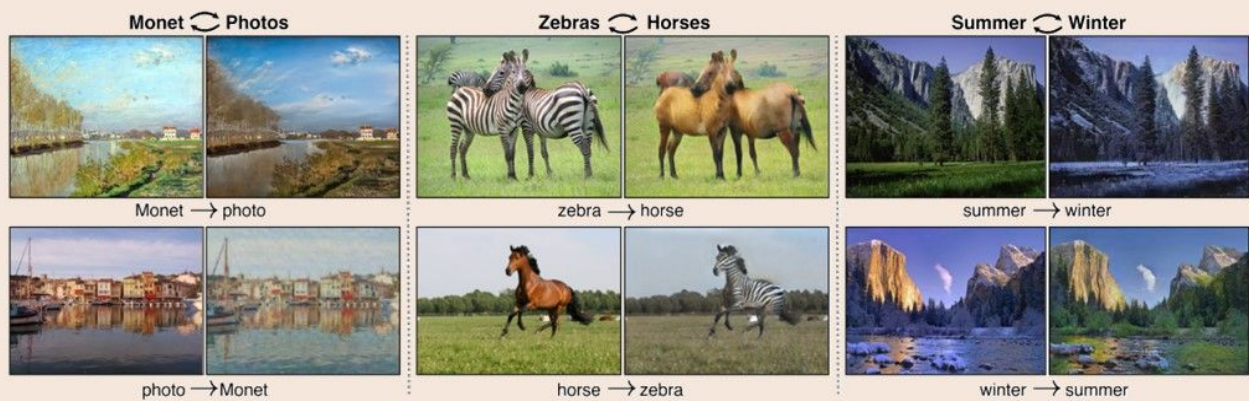
진짜 데이터 \mathbf{x} 를 넣었을 때의 값 G 가 만든 가짜 데이터 $G(\mathbf{z})$ 를 넣었을 때의 값

— : 진짜 데이터를 넣으면 큰 값, 가짜 데이터를 넣으면 작은 값을 출력하도록 Discriminator를 학습시키자!

— : $D(G(\mathbf{z}))$ 를 극대화하는 G , 즉 Discriminator를 속이는 Generator를 학습시키자!

1. CycleGAN(Cycle Generative Adversarial Network)

: 각각 2개의 Generator와 Discriminator를 이용해 전체적인 형태를 유지하면서 스타일만 변형해주는 GAN

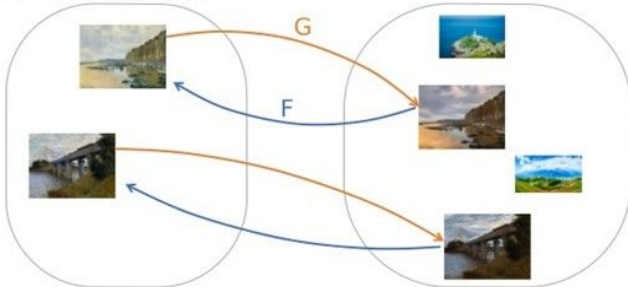


1. CycleGAN(Cycle Generative Adversarial Network)

Cycle Consistency

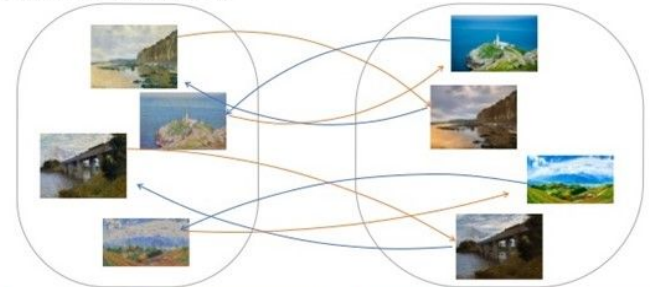
이제 CycleGAN은 Discriminator를 이용해 전체적인 형태를 유지하면서 스타일만 변형해주는 GAN

CycleGAN (Zhu et al. 2017)



$$L_{GAN}(G(x), y) + \|F(G(x)) - x\|_1$$

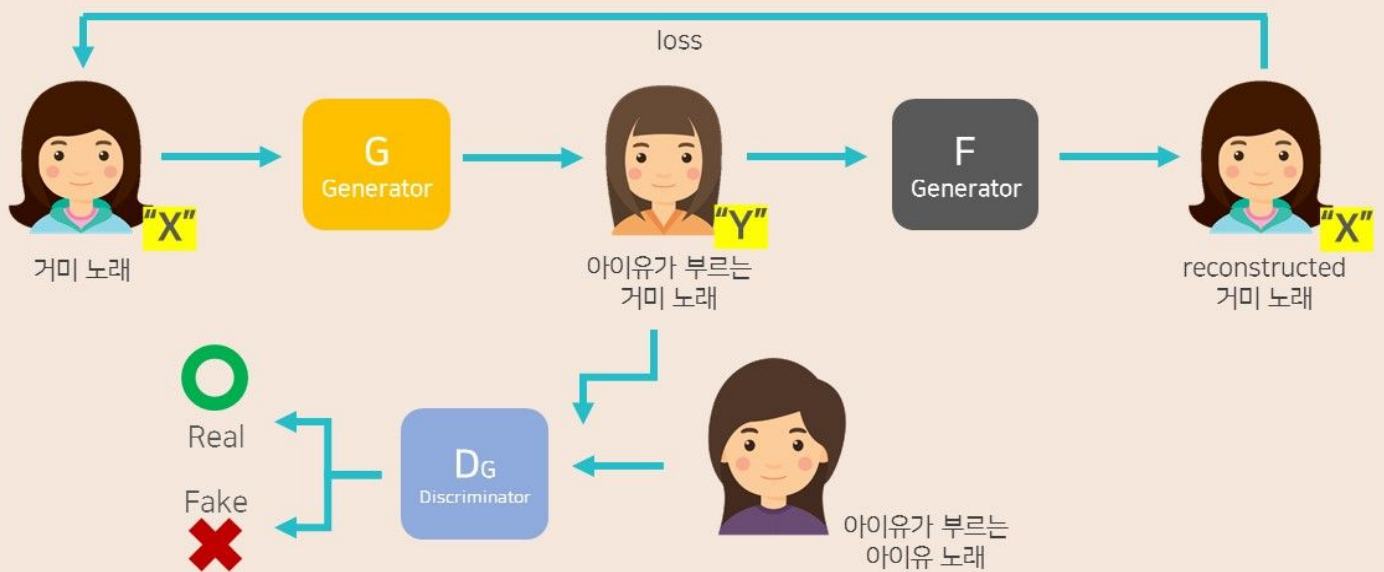
CycleGAN (Zhu et al. 2017)



$$L_{GAN}(G(x), y) + \|F(G(x)) - x\|_1 + L_{GAN}(F(y), x) + \|G(F(y)) - y\|_1$$

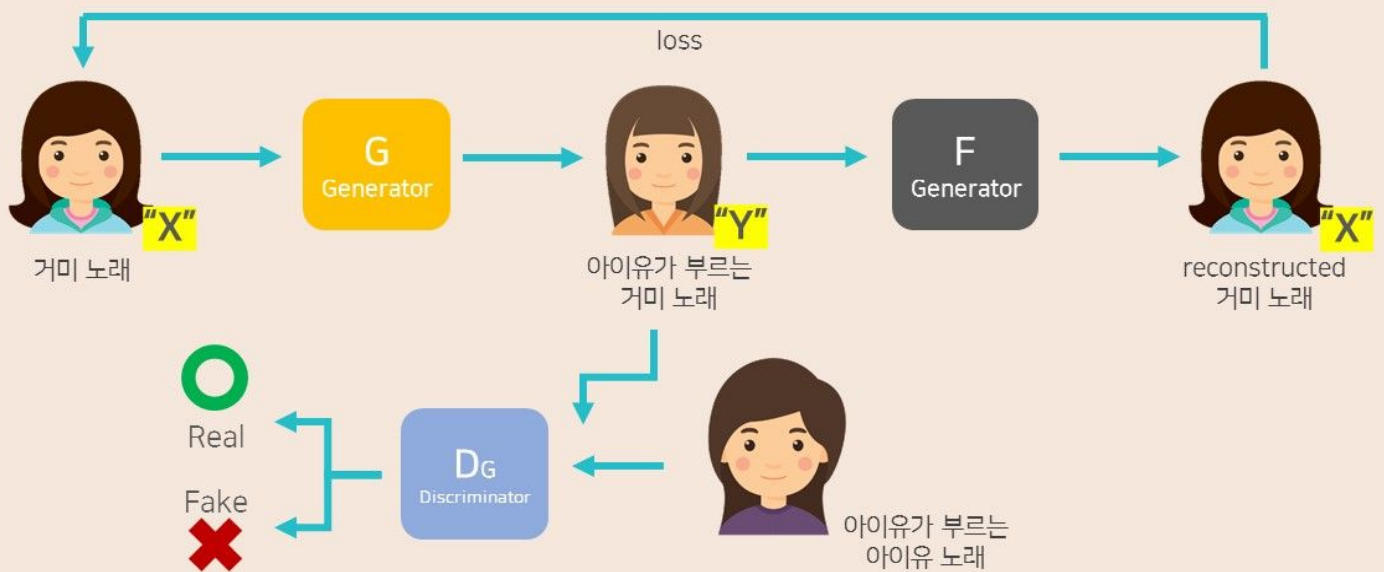
1. CycleGAN(Cycle Generative Adversarial Network)

: 각각 2개의 Generator와 Discriminator를 이용해 전체적인 형태를 유지하면서 스타일만 변형해주는 GAN



1. CycleGAN(Cycle Generative Adversarial Network)

: 각각 2개의 Generator와 Discriminator를 이용해 전체적인 형태를 유지하면서 스타일만 변형해주는 GAN



1. CycleGAN(Cycle Generative Adversarial Network) - Loss Function

: 각각 2개의 Generator와 Discriminator를 이용해 전체적인 형태를 유지하면서 스타일만 변형해주는 GAN

$$\mathcal{L}_{full} = \mathcal{L}_{adv}(G_{X \rightarrow Y}, D_Y) + \mathcal{L}_{adv}(G_{Y \rightarrow X}, D_X) + \lambda_{cyc} \mathcal{L}_{cyc}(G_{X \rightarrow Y}, G_{Y \rightarrow X}) + \mathcal{L}_{id}(G_{X \rightarrow Y}, G_{Y \rightarrow X})$$



$$\mathcal{L}_{adv}(G_{X \rightarrow Y}, D_Y) = \mathbb{E}_{y \sim P_{Data}(y)} [\log D_Y(y)] + \mathbb{E}_{x \sim P_{Data}(x)} [\log(1 - D_Y(G_{X \rightarrow Y}(x)))]$$

GAN Loss: GAN의 Generator와 Discriminator를 적대적으로 학습하기 위한 Loss이다.
Generator는 X 도메인에서 Y 도메인과 비슷한 데이터를 생성하고,
Discriminator는 진짜 Y도메인 데이터인지 판별하며 학습한다.

1. CycleGAN(Cycle Generative Adversarial Network) - Loss Function

: 각각 2개의 Generator와 Discriminator를 이용해 전체적인 형태를 유지하면서 스타일만 변형해주는 GAN

$$\mathcal{L}_{full} = \mathcal{L}_{adv}(G_{X \rightarrow Y}, D_Y) + \mathcal{L}_{adv}(G_{Y \rightarrow X}, D_X) + \lambda_{cyc} \mathcal{L}_{cyc}(G_{X \rightarrow Y}, G_{Y \rightarrow X}) + \mathcal{L}_{id}(G_{X \rightarrow Y}, G_{Y \rightarrow X})$$



$$\mathcal{L}_{cyc}(G_{X \rightarrow Y}, G_{Y \rightarrow X}) = \mathbb{E}_{x \sim P_{Data}(x)} [\|G_{Y \rightarrow X}(G_{X \rightarrow Y}(x)) - x\|_1] + \mathbb{E}_{y \sim P_{Data}(y)} [\|G_{X \rightarrow Y}(G_{Y \rightarrow X}(y)) - y\|_1]$$

Cycle Consistency Loss: Generator에 의해 생성된 가짜 데이터를
다시 반대 도메인으로 생성한 데이터와
기존 원본 데이터의 Loss이다.

1. CycleGAN(Cycle Generative Adversarial Network) – Loss Function

: 각각 2개의 Generator와 Discriminator를 이용해 전체적인 형태를 유지하면서 스타일만 변형해주는 GAN

$$\mathcal{L}_{full} = \mathcal{L}_{adv}(G_{X \rightarrow Y}, D_Y) + \mathcal{L}_{adv}(G_{Y \rightarrow X}, D_X) + \lambda_{cyc} \mathcal{L}_{cyc}(G_{X \rightarrow Y}, G_{Y \rightarrow X}) + \mathcal{L}_{id}(G_{X \rightarrow Y}, G_{Y \rightarrow X})$$

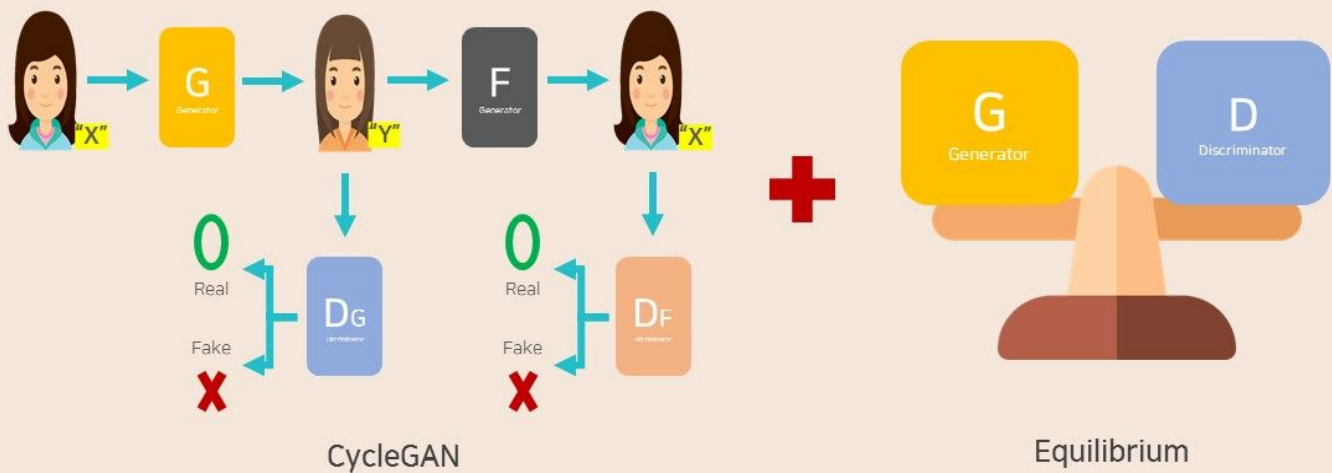


$$\mathcal{L}_{id}(G_{X \rightarrow Y}, G_{Y \rightarrow X}) = \mathbb{E}_{y \sim P_{Data}(y)} [\|G_{X \rightarrow Y}(y) - y\|_1] + \mathbb{E}_{x \sim P_{Data}(x)} [\|G_{Y \rightarrow X}(x) - x\|_1]$$

Identity Loss: 진짜 Y 도메인의 데이터가 Y 도메인 Discriminator에 들어왔을 때의 Mapping Function.
이 과정에서 Y 도메인의 특징을 유지하도록 만들어
X 도메인의 형태를 깊게 학습할 수 있도록 한다!

2. CycleGAN + BEGAN(Boundary Equilibrium Generative Adversarial Network)

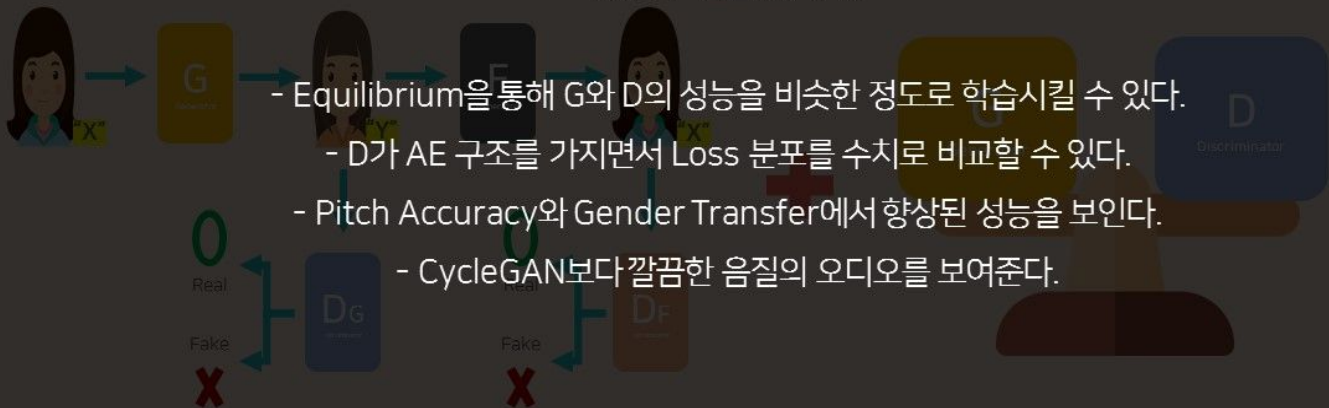
: CycleGAN과 동일한 구조에서 D가 AutoEncoder를 가지며 Equilibrium을 통해 G와 D의 평형을 유지하는 GAN



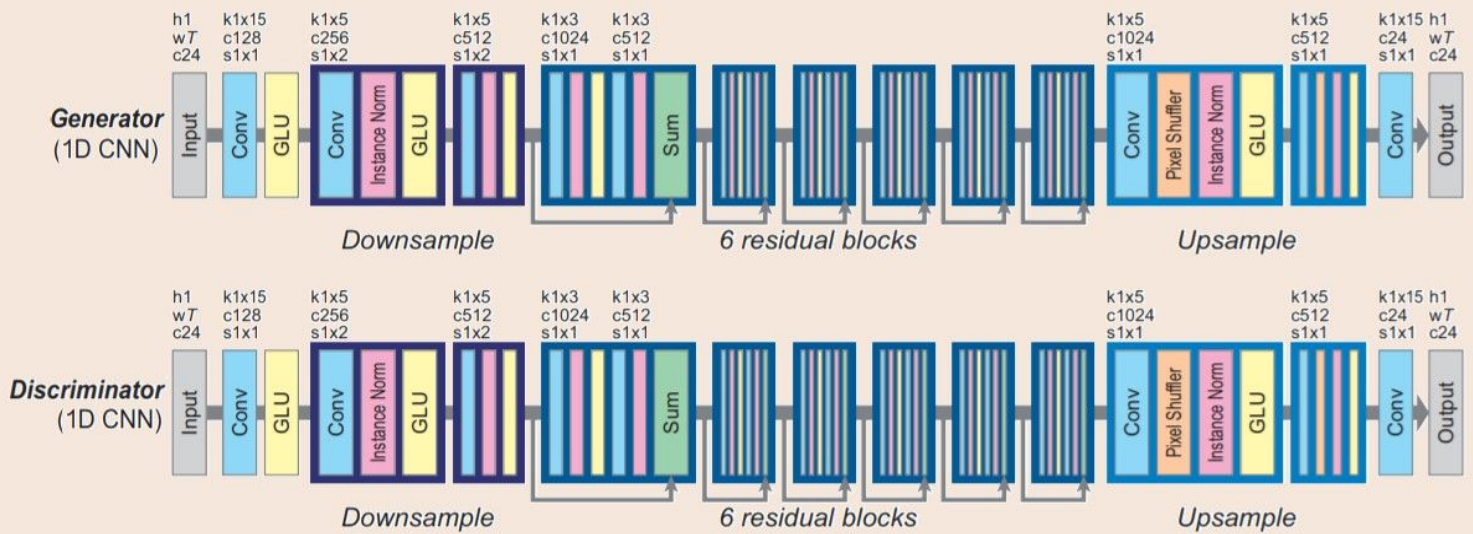
2. CycleGAN + BEGAN(Boundary Equilibrium Generative Adversarial Network)

: CycleGAN과 동일한 구조에서 D가 AutoEncoder를 가지며 Equilibrium을 통해 D와 G의 평형을 유지하는 GAN

WHY BEGAN?



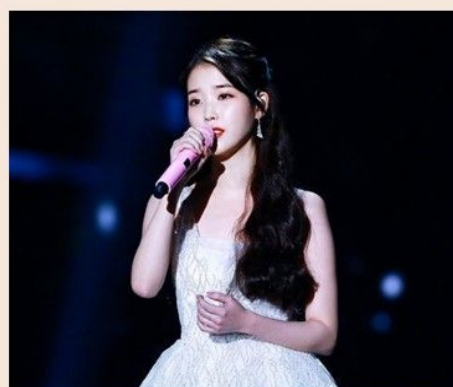
3. Full Architecture



05. Conclusion

Final Result & Discussion

1. 결과 시연 : 거미 - You are My Everything(아이유 Ver.)



1. 결과 시연 : 케이윌 - 내 생애 아름다운(10cm Ver.)



2. 한계점



apple → orange



photo → Van Gogh

Singing Style(창법) Conversion은 불가능, Singing Voice Conversion만 가능 ☹

2. 한계점



Hyper-parameter Tuning ☹

~~n_frames~~

~~frame_period~~

~~epoch~~

~~cycle_gamma~~

데이터 변경 & 하이퍼 파라미터 튜닝을 시도했지만
가시적인 변화는 보지 못 함

3. 개선 방향

- WaveNet AutoEncoder를 적용한 시도

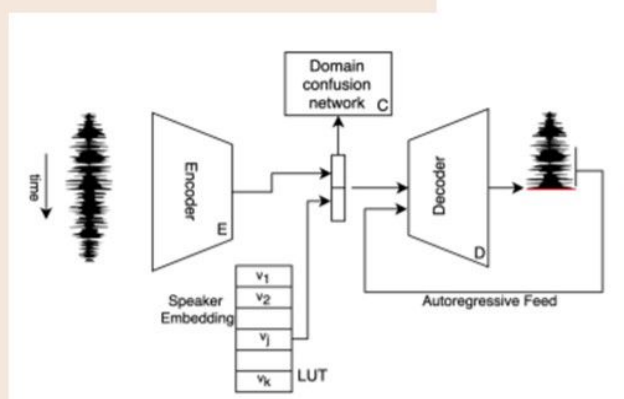
Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders

ck^{*1} Adam Roberts¹ Sander Dieleman² Douglas Eck¹
Simonyan² Mohammad Norouzi¹

seen rapid
rovements
ity image
contribute
de similar
it, we de-
ie autoen-
utoregres-

perhaps because of them, synthesizers have had a pro-
found effect on the course of music and culture in the
past half century (Punk, 2014).

In this paper, we outline a data-driven approach to
audio synthesis. Rather than specifying a specific ar-
rangement of oscillators or an algorithm for sample
playback, such as in FM Synthesis or Granular Synthe-
sis (Chowning, 1973; Xenakis, 1971), we show that it is
possible to generate new types of expressive and real-
istic instrument sounds with a neural network model.





김유민
투빅스 11기
서울시립대학교
영어영문학과
16학번



이소라
투빅스 11기
덕성여자대학교
정보통계학과
15학번



정혜인
투빅스 11기
숙명여자대학교
컴퓨터과학전공
17학번



박진혁
투빅스 12기
성균관대학교
전기전자공학부
15학번



조민호
투빅스 12기
서강대학교
철학과
14학번



최승호
투빅스 12기
국민대학교
소프트웨어학부
14학번



감사합니다 😊

김유민

박진혁

이소라

정혜인

조민호

최승호