Generative Adversarial Networks for Voice Conversion

Chao WANG
CentraleSupelec
Gif-sur-Yvette, France
chao.wang@student-cs.fr

February 27, 2021

Abstract

CycleGAN-VC3[11] is a generative adversarial network model published in 2020. It takes mel-spectrogram as input and converts a person's speech to another person's voice. Since its code is not officially open-sourced, this project is its effective implementation, and evaluation is based on the VCC 2020 dataset[13].

1. Motivation

Voice conversion (VC) refers to digital cloning of a person's voice; it can be used to modify audio waveform so that it appear as if spoken by someone else (target) than the original speaker (source). VC is useful in many applications, such as customizing audio book and avatar voices, dubbing, movie industry, teleconferencing, singing voice modification, voice restoration after surgery, and cloning of voices of historical persons. Since VC technology involves identity conversion, it can also be used to protect the privacy of the individual in social media and sensitive interviews, for instance.

2. Problem Definition

Voice conversion is a technique for learning mappings between source and target speeches, for example the conversion of a male's speech to a female's voice. However, the collection of parallel corpus is often impractical, and even if such data is collected, the time alignment procedure remains a challenge. Thus, the technique needs to be trained with non-parallel data, which meets perfectly generative adversarial networks (GANs)'s advantage. Thus, our project will train and evaluate a recently developed model cycleGAN-VC3[11] 's voice conversion performance on the Voice Conversion Challenge 2020 database[13]'s intralingual non-parallel dataset. There is no official open-source code, only 1 reproduced code which is incomplete and has some inconsistence and problem compared with the paper. So, this project is an efficient implementation of the deep learning algorithm and experimental evaluation.

3. Related Work

For non-parallel voice conversion problem, many recurrent neural network (RNN) and generative adversarial network (GAN) are proposed, such as deep bidirectional long short term memory (DBLSTM) based recurrent neural network (RNN)[2], Wasserstein generative adversarial network (WDGAN)[14], and StarGAN[7].

Recently, encoder-decoder based networks[4], such as variational autoencoder (VAE) [5], variational autoencoding Wasserstein generative adversarial network (VAW-GAN) [6], auxiliary classifier variational autoencoder (ACVAE) [8], and cycle-consistent autoencoder (CycleVAE)[12] have also been successfully applied into various tasks, such as cross-lingual voice conversion and emotional voice conversion. But VAE is based on the assumption that source and target speech lie in the same low-dimensional embeddings. This would cause difficulty in modeling complex structures, e.g., detailed spectrotemporal structure. So in this project, instead of using VAE, we will use the GAN structure, which learn a mapping function directly without embedding.

4. Methodology

4.1. From waveform to mel-frequency ceptral coefficients

1. Spectrum

   The original audio file could be downsampled to a time series, and we could use Discrete Fourier Transform to get this entire time series' frequency content, which is a 1D spectrum, as each number is the a specified

frequency's altitude of this signal.

$$\widehat{x_{(\frac{k}{N})}} = \sum_{n=0}^{N-1} x_{(n)} e^{-i2\pi n \frac{k}{N}}$$
$$k \in [0, M-1 = N-1]$$
$$\text{by taking } \#\text{frequencies } (M) = \#\text{samples } (N)$$

2. Spectrogram

   But since most audio signals such as music and speech are known as non periodic signals, we want to know the variation of spectrum over time. Short-Time Fourier Transform is a way to realize this, by applying a windowing function w (usually the Hanning window) to the signal, and do Discrete Fourier Transform to every short time period of signal, so its output is a 2D matrix, called spectrogram.

$$S_{(m,k)} = \sum_{n=0}^{N-1} x_{(n+mH)} w_{(n)} e^{-i2\pi n \frac{k}{N}}$$

   m: the current time frame, k: the specified frequency bin
   H: hop size, $n \in [0, N-1]$: all samples in that frame

3. Mel spectrogram

   Studies have shown that humans do not perceive frequencies on a linear scale. We are better at detecting differences in lower frequencies than higher frequencies. For example, we can easily tell the difference between 500 and 1000 Hz, but we will hardly be able to tell a difference between 10,000 and 10,500 Hz, even though the distance between the two pairs are the same.

   In 1937, Stevens & al. proposed a unit of pitch such that equal distances in pitch sounded equally distant to the listener. This is called the mel scale. The mathematical equation of converting frequency to mel scale is:

$$m = 2595 \log_{10}\left(1 + \frac{f}{500}\right)$$

   By choosing the number of mel bands, constructing mel filter banks, and applicating mel filter banks to the spectrogram in dB, we could get the 2D mel spectrogram, whose one dimension is still # time frames, but the other dimension is no longer # frequency bins, but rather # mel bands.

4. Ceptrum

   The term cepstrum was derived by reversing the first four letters of spectrum, since ceptrum is the result of computing the Inverse Discrete Fourier Transform of the logarithm of spectrum, which is an 1D array gotten from the Discrete Fourier Transform $\mathcal{F}$ as presented above.

$$C_{x_{(t)}} = \mathcal{F}^{-1}\{\log(\mathcal{F}\{x_{(t)}\})\}$$

   Ceptrum is an 1D array constituted of the absolute magnitude of different quefrecies (quenfrecy : reversing the first four letters of frequency, unit : second since it is homogenous to a time). By applicating a low-pass lifter (lifter : reversing the first three letters of filter), we could get the signal's several first rhamonics (rhamonic : reversing the first three letters of harmonic). The meaning of these first rhamonics is similar to the formants in the spectral envelope : they carry the "large" structures (identity such as pitch) of spectrum and ignore the "fine" structures.

5. Mel-frequency ceptral coefficients

   Since spectrum's frequencies' difference is different from human's perception, so we first applicate the mel-scaling to the logarithm of spectrum. Then, instead of doing the Inverse Discrete Fourier Transform, we do Discrete Cosine Transorm, since it's simpler, its coefficients are real instead of complex, and it could decorrelate the different mel bands. The output 1D array is called mel-frequency ceptral coefficients (MFCCs).

   As a low-pass lifter, we choose the first 13 coefficients for example, and we also take into account the first derivative and second derivative of MFCCs, so there are 13*3=39 coefficients in total.

   If the Fourier Transform used is Short-Time Fourier Transform instead of Discrete Fourier Transform , then the output will be a 2D matrix: 1 dimension is # time frames, the other dimension is # MFCCs, which means that each time frame has 39 coefficients.

The objective of the experiment is to analyze the quality of the converted MFCCs.

4.2. CycleGAN

The goal of CycleGAN [15] is to learn 2 Generators $G : X \to Y$ and $F : Y \to X$ between two domains X and Y given training samples $\{x_i\}_{i=1}^{N}$ where $x_i \in X$ and $\{y_j\}_{j=1}^{M}$ where $y_j \in Y$. We denote the data distribution as $x \sim P_X$ and $y \sim P_Y$. As shown in Figure 1, Generator G is equipped with a Discriminator D_Y that aims to discriminate between $\{y\}$ and $\{G(x)\}$, and Generator F is equipped with a Discriminator D_X that aims to distinguish between images $\{x\}$ and translated images $\{F(y)\}$.

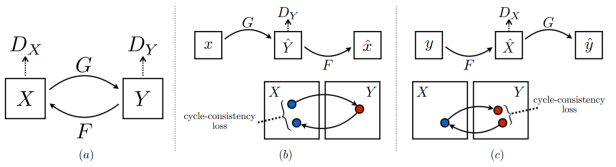

Figure 1. CycleGAN

The objective contains 2 types of losses:

1. Adversarial Losses: It is applied to each Generator-Discriminator pair. For Generator $G : X \to Y$ and its Discriminator D_Y, we express the adversarial loss as:

$$\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim P_Y(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim P_X(x)}[\log(1 - D_Y(G(x))]$$

where G tries to generate images $G(x)$ that look similar to images from domain Y, while D_Y aims to distinguish between translated samples $G(x)$ and real samples y. G aims to minimize this objective against its adversary D who tries to maximize this objective, i.e., $\min_G \max_{D_Y} \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)$. Similarly, the adversarial loss for Generator $F : Y \to X$ and its Discriminator D_X is: $\min_F \max_{D_X} \mathcal{L}_{\text{GAN}}(F, D_X, Y, X)$.

2. Cycle Consistency Loss: To make sure that the generated output still conserves input's information and is not just a random sample from the desired domain, the learned Generators should be cycle-consistent: for each $x \in X$, the image translation cycle should be able to bring x back to the original x, i.e., $x \to G(x) \to F(G(x)) \approx x$, we call this forward-cycle consistency. Similarly, for each $y \in Y$, F and G should also satisfy backward cycle consistency: $y \to F(y) \to G(F(y)) \approx y$. We incentivize this behavior using a cycle consistency loss:

$$\mathcal{L}_{\text{cyc}}(G, F) = \mathbb{E}_{x \sim P_X(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim P_Y(y)}[\|G(F(y)) - y\|_1]$$

Thus, the full objective is:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F)$$

where λ controls the relative importance of the two objectives. We aim to solve:

$$G^*, F^* = \arg \min_{G,F} \max_{D_X, D_Y} \mathcal{L}(G, F, D_X, D_Y)$$

Although the intention of cycle consistency is good, in fact it has a steganography problem [3]: the Generator G's output $G(x)$ might invisibly includes some information about the input in a low-amplitude but high-frequency signal, which makes $G(x)$ dissimilar to the input x but the Generator F could still restore the original input.

4.3. CycleGAN-VC 1∼3

CycleGAN-VC[9] and CycleGAN-VC2[10] used CycleGAN's idea on the parallel-data-free voice conversion problem, for which $X = \mathbb{R}^{Q \times T_x}$, $Y = \mathbb{R}^{Q \times T_y}$, where Q is the feature dimension, T_x and T_y are the sequence lengths.

They improve CycleGAN in 2 aspects: the objective function and the neural network architecture.

4.3.1 Objective function

First, they add 1 identity-mapping loss and 2 two-step adversarial losses to the objective function:

1. Identity-mapping loss: To further encourage the preservation of input's linguistic information, CycleGAN-VC[9] adds a this loss term:

$$\mathcal{L}_{id}(G, F) = \mathbb{E}_{y \sim P_Y(y)}[\|G(y) - y\|_1] + \mathbb{E}_{x \sim P_X(x)}[\|F(x) - x\|_1]$$

which encourages the generator to find the mapping that preserves composition between the input and output.

2. Two-Step Adversarial Losses: To mitigate the over-smoothing problem caused by cycle-consistency loss, CycleGAN-VC2[10] introduces an additional Discriminator D'_X to distinguish in domain X the real x and the $F(G(x))$ generated by the forward cycle (Generator G then Generator F), so its loss is:

$$\mathcal{L}_{adv2}(G, F, D'_X) = \mathbb{E}_{x \sim P_X(x)}[\log D'_X(x)] + \mathbb{E}_{x \sim P_X(x)}[\log(1 - D'_X(F(G(x))))]$$

Similarly, for the backward cycle, the adversarial loss for Discriminator D'_Y is $\mathcal{L}_{adv2}(F, G, D'_Y)$.

For each loss term, a trade-off parameter λ is added as its weight.

4.3.2 Neural network architecture

1. Gated-CNN

In order to represent the sequential and hierarchical structures of speech and to allow parallelization over sequential data at the same time, instead of using RNN, CycleGAN-VC[9] uses a 1D gated CNN whose activation function is gated linear units (GLUs). A GLU is a data-driven activation function, and the $(l+1)$ -th layer output $\boldsymbol{H}_{l+1}$ is calculated using the l -th layer output $\boldsymbol{H}_l$ and model parameters $\boldsymbol{W}_l$, $\boldsymbol{V}_l$, $\boldsymbol{b}_l$, and $\boldsymbol{c}_l$,

$$\boldsymbol{H}_{l+1} = (\boldsymbol{H}_l * \boldsymbol{W}_l + \boldsymbol{b}_l) \otimes \sigma(\boldsymbol{H}_l * \boldsymbol{V}_l + \boldsymbol{c}_l)$$

where $\otimes$ is the element-wise product and σ is the sigmoid function. This gated mechanism allows the information to be selectively propagated depending on the previous layer states.

2. Generator : 2-1-2D CNN

For the generator, a 1D CNN (used in CycleGAN-VC[9]) is more feasible for capturing dynamical change, as it can capture the overall relationship along with the feature dimension. In contrast, a 2D CNN is better suited for converting features while preserving the original structures, as it restricts the converted region to local. Even using a 1D CNN, residual blocks can mitigate the loss of the original structure, but downsampling and upsampling (which are necessary for effectively capturing the wide-range structures) become a severe cause of this degradation.

To alleviate it, CycleGAN-VC2[10] has developed a network architecture called a 2-1-2D CNN, shown in Figure 2. In this network, 2D convolution is used for downsampling and upsampling, and 1D convolution is used for the main conversion process (i.e., residual blocks). To adjust the channel dimension, 1×1 convolution is applied before or after reshaping the feature map.

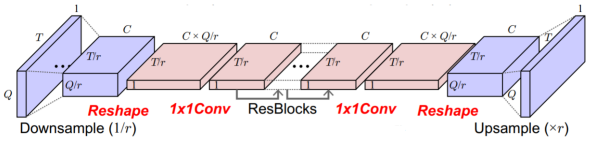

Figure 2. 2-1-2D CNN

3. TFAN in 2-1-2D CNN

CycleGAN-VC3 [11] discovered that the previous generator compromises mel-cepstrum's time-frequency structure that should be preserved in conversion, so they devise a part of neural network named TFAN to replace the instance normalization (IN) in 2-1-2D CNN Generator.

Given feature f, TFAN normalizes it in a channel-wise manner similar to IN, and then modulates the normalized feature in an element-wise manner using scale $\gamma(\boldsymbol{x})$ and bias $\boldsymbol{\beta}(\boldsymbol{x})$, which are calculated from $\boldsymbol{x}$ using multi-layer CNNs:

$$f' = \gamma(x)\frac{f-\mu(f)}{\sigma(f)} + \beta(x)$$

where f' is the output feature, and $\mu(f)$ and $\sigma(f)$ are the channel-wise average and standard deviation of f, respectively. Its neural network representation is Figure 3.

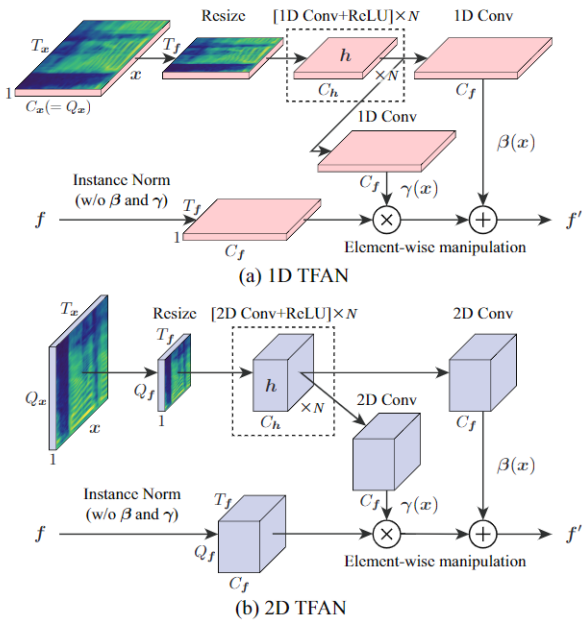

Figure 3. TFAN

In IN, scale β and bias γ are x -independent and are applied in a channel-wise manner, whereas in TFAN, they are calculated from $\boldsymbol{x}$ and are applied in an element-wise manner. These differences allow TFAN to adjust the scale and bias of f while reflecting x in a time- and frequency-wise manner.

4. Discriminator : PatchGAN

CycleGAN-VC2[10] replaced FullGAN with PatchGAN 4, which uses convolution rather than fully connection at the last layer, and thus determines the realness on the basis of the patch, rather than a single scalar output which signifies "real" or "fake".

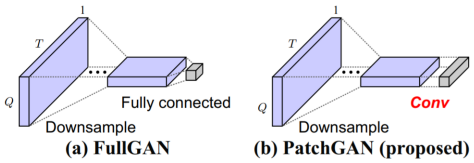

Figure 4. Comparison of FullGAN and PatchGAN

This reduces the parameters and stabilizes the GAN training.

So CycleGAN-VC2's total netowrk architecture is Figure 5.

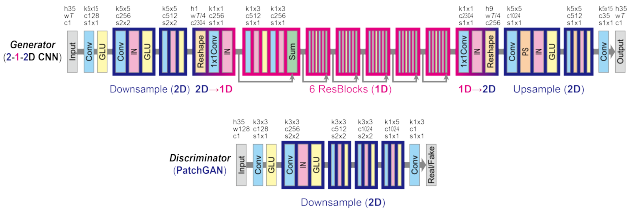

Figure 5. CycleGAN-VC2

And in CycleGAN-VC3, the IN in the 1D→2D block and that in the upsampling block were replaced with 1D TFAN and 2D TFAN respectively, as in Figure 6.

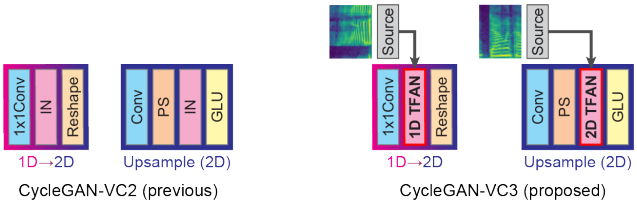

Figure 6. TFAN replace IN

5. Evaluation

5.1. Evaluation metrics

Since source person's speech and target person's speech could not be absolutely aligned even if their speech content are the same, direct measurement of the difference between the target and converted mel-spectrograms is difficult. As an alternative, the two following evaluation metrics are used to evaluate the model's performance in voice conversion:

1. Mel cepstral distortion (MCD) : the root mean squared error against the ground truth signal, computed on 13 dimensional MFCCs using dynamic time warping[1] to align features computed on the synthesized audio to those computed from the ground truth. MFCC features are computed from an 80-channel log-mel spectrogram using a 50ms Hann window and hop of 12.5ms.

$$MCD\left(v^{\text{targ}}, v^{\text{ref}}\right) = \frac{\alpha}{T}\sum_{t=0}^{T-1}\sqrt{\sum_{d=s}^{D}\left(v_d^{\text{targ}}(t) - v_d^{ref}(t)\right)^2}$$

$$\alpha = \frac{10\sqrt{2}}{\ln 10} = 6.14185$$

2. Mel spectral distortion (MSD) : the same as MCD but applied to the log-mel spectrogram magnitude instead of cepstral coefficients. This captures harmonic content which is explicitly removed in MCD due to liftering, so assesses the local structural difference.

For both metrics, the smaller the value, the better the performance.

5.2. Implementation structure

As an implementation of CycleGAN-VC3[11], our program is strctured as following:

1. preprocess_training.py : First, convert audio file to waveform, with the downsampling rate 22050 Hz. Second, use pre-trained melgan vocoder to transform the waveform to 80-dimensional mel-spectrogram. Third, normalize the logarithm of mel-spectrogram for source dataset and target dataset separately. Forth, randomly shuffle the dataset, then extract 64 time frames from each mel-spectrogram. The reason of enforcing mel-spectrograms to have the same time frames for training stage is that the computation of loss needs its 2 input mel-spectrograms to be definitely at the same size in order to compare reasonably. In contrast, since all computation used in generator is convolution, it could take any time-frames' mel-spectrogram as input, which means it could convert any audio file to another voice for evaluation and testing stage. Last, save the training dataset gotten, as well as the mean and standard deviation of source dataset and target dataset to cache folder, since they will be used for generating the converted audio files in evaluation and testing later.

2. smallmodulles.py : wrap up down-samplining, upsampling, GLU, pixel shuffling, and residual block as classes, which will be used in building the generator and discriminator later.

3. generatordiscriminator.py : build generator and discriminator classes, according to the figure 5.

4. trainclass.py : define the class for training, which includes the calculation of loss, and the voice conversion of validation sets.

5. train.py : the execution python file for training.

6. evaluation.py : with the converted voice folder and the target voice folder, calculate the mean and standard deviation of mcd and msd score, and save the comparison of their waveform, fundamental frequency, mel-cepstral coefficients, and aperiodicity's comparison images.

The running order should be : preprocesstraining.py, train.py, evaluation.py.

5.3. Results

We evaluate the built CycleGAN-VC3 using the Voice Conversion Challenge 2020 database[13]. The model is trained on colab's GPU for 50,000 iterations (2 days), with the batch size equals to 1.

The MCD and MSD score for different tasks are presented in table 1.

Table 1. MCD MSD

Task	MCD	MSD
A generate B	10.67±0.33	0.95±0.05
B generate A	11.18±0.38	0.96±0.07

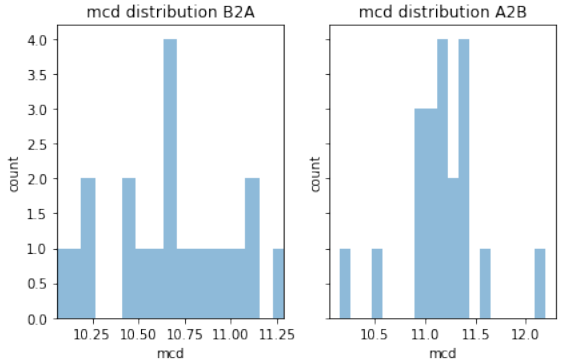

Figure 7. MCD

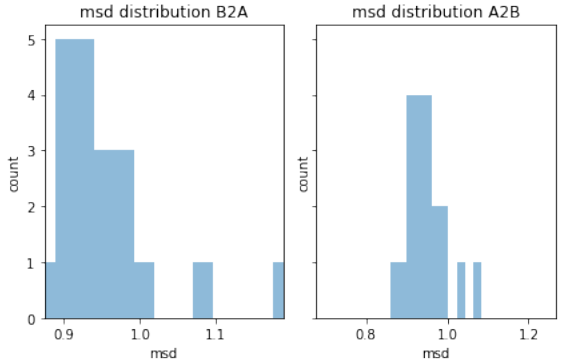

Figure 8. MSD

We could also compare in detail the waveform and mel-ceptrum of the source, converted, and target audios, which give us a direct impression of their similarity and difference. Figure 9 and figure 10 are waveform and mel-ceptrum respectively. In each figure, the 3 rows' speech content are the same, while the first row is the source, the second row is the converted one, and the third row is the target. We could find that for the converted one, some of its source characteristics have disappeared, and some target characteristics have been added.

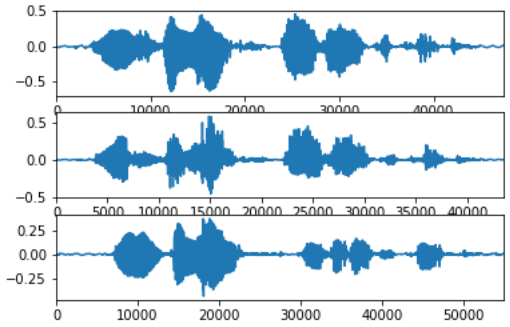

Figure 9. waveform

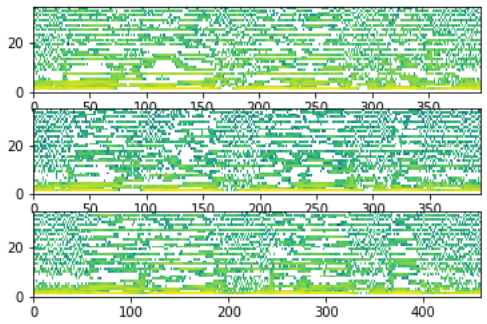

Figure 10. Mel-ceptrum

Subjective evaluation is about inviting multipal listeners to really listen the different audios and give score on speaker similarity, which is not practical for me. But readers could listen to some converting examples at this link.

6. Conclusion and Discussion

For both MCD and MSD , the mean is much higher than the standard deviation, which means the mean is quite reliable. My experiment's MCD is a little larger than the paper's result, with paper's MCD is 7∼9. But MY experiment's MSD is the very similar to the paper's result. The difference is might because of the paper trained for 500,000 iterations, while I trained 50,000 iterations with limited resources.

This project realized an effective implementation of CycleGAN-VC3[11], which could be used directly for end-to-end speech voice conversion. Since I spent a lot of time building the program and the computation resources are limited, I didn't compare its performance with other algorithms, but this program could be used for comparison in the future. Moreover, attention mechanism could be further integrated to ameliorate the model.

References

[1] Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, AAAIWS'94, page 359–370. AAAI Press, 1994.

[2] Xin Chen, Wei Chu, Jinxi Guo, and Ning Xu. Singing voice conversion with non-parallel data. In *2nd IEEE Conference on Multimedia Information Processing and Retrieval, MIPR 2019, San Jose, CA, USA, March 28-30, 2019*, pages 292–296, 2019.

[3] Casey Chu, Andrey Zhmoginov, and Mark Sandler. Cyclegan, a master of steganography, 2017.

[4] Gustav Eje Henter, Jaime Lorenzo-Trueba, Xin Wang, and Junichi Yamagishi. Deep encoder-decoder models for unsupervised learning of controllable speech synthesis, 2018.

[5] Chin-Cheng Hsu, Hsin-Te Hwang, Yi-Chiao Wu, Yu Tsao, and Hsin-Min Wang. Voice conversion from non-parallel corpora using variational auto-encoder, 2016.

[6] Chin-Cheng Hsu, Hsin-Te Hwang, Yi-Chiao Wu, Yu Tsao, and Hsin-Min Wang. Voice conversion from unaligned corpora using variational autoencoding wasserstein generative adversarial networks, 2017.

[7] Hirokazu Kameoka, Takuhiro Kaneko, Kou Tanaka, and Nobukatsu Hojo. Stargan-vc: Non-parallel many-to-many voice conversion with star generative adversarial networks, 2018.

[8] Hirokazu Kameoka, Takuhiro Kaneko, Kou Tanaka, and Nobukatsu Hojo. Acvae-vc: Non-parallel many-to-many voice conversion with auxiliary classifier variational autoencoder, 2020.

[9] Takuhiro Kaneko and Hirokazu Kameoka. Parallel-data-free voice conversion using cycle-consistent adversarial networks, 2017.

[10] Takuhiro Kaneko, Hirokazu Kameoka, Kou Tanaka, and Nobukatsu Hojo. Cyclegan-vc2: Improved cyclegan-based non-parallel voice conversion, 2019.

[11] Takuhiro Kaneko, Hirokazu Kameoka, Kou Tanaka, and Nobukatsu Hojo. Cyclegan-vc3: Examining and improving cyclegan-vcs for mel-spectrogram conversion. In *Proceedings of the Annual Conference of the International Speech Communication Association*, 2020.

[12] Patrick Lumban Tobing, Yi-Chiao Wu, Tomoki Hayashi, Kazuhiro Kobayashi, and Tomoki Toda. Non-parallel voice conversion with cyclic variational autoencoder, 2019.

[13] Zhao Yi, Wen-Chin Huang, Xiaohai Tian, Junichi Yamagishi, Rohan Kumar Das, Tomi Kinnunen, Zhen-Hua Ling, and Tomoki Toda. Voice Conversion Challenge 2020 — Intra-lingual semi-parallel and cross-lingual voice conversion —. In *Proc. Joint Workshop for the Blizzard Challenge and Voice Conversion Challenge 2020*, pages 80–98, 2020.

[14] W. Zhao, W. Wang, Y. Sun, and T. Tang. Singing voice conversion based on wd-gan algorithm. In *2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, volume 1, pages 950–954, 2019.

[15] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.