Voice Conversion by using CycleGAN-VC (January 2021)

Selahaddin HONİ, İsmail Melik TÜRKER and İmran Çağla EYÜBOĞLU

***Abstract*—In the project, it is aimed to transfer the trained voice style of a famous person to given input voice. A reference paper, "*Parallel-Data-Free Voice Conversion Using Cycle-Consistent Adversarial Networks*" published by Takuhiro Kaneko and Hirokazu Kameoka in 2017, was selected to realize this purpose. They participated into Voice Conversion Challenge 2016 with their work; this paper demonstrates an adaptation of that work to Turkish language. A famous female news-presenter is chosen as target voice and we tried to transfer this voice style to both male and female source voices. Synthesized fake voices are uploaded to deployed project web page.**

***Index Terms*—Voice Conversion, CycleGAN, Turkish linguistic style, VC Challenge 2016.**

***Project Page*—A project publish page is deployed for the results, related links and more:**

mlsp2020.github.io

I. Introduction

AIM of this project is manipulating and generating some famous person's voice by using voice conversion (VC) techniques.

Main motivation behind VC is extracting the feature of source voice and regenerate an output to target form (as another target voice or text etc.). One of the most common application of VC is speech to text or text to speech conversion. In our study we will try to use VC techniques and generate a famous person's voice. Some of these techniques are Codebook mapping, Vector Quantization, Dynamic Frequency Warping, Hidden Markov Models, Gaussian Mixture Models and Artificial Neural Networks.

After Ezzine and Frikha compared the techniques applied on voice conversion in their paper, they state as a conclusion that "The perceptual tests shown that techniques based on neural network provide better output quality even in the presence of noise once the number of layers increases in the neural network. More precisely, to obtain greater synthesis accuracy we should essentially get a deeper non-linear architecture with lots of hidden layers." [1] Thus, utilizing neural networks, particularly a new era: Generative Adversarial Networks, is expected to give better results.

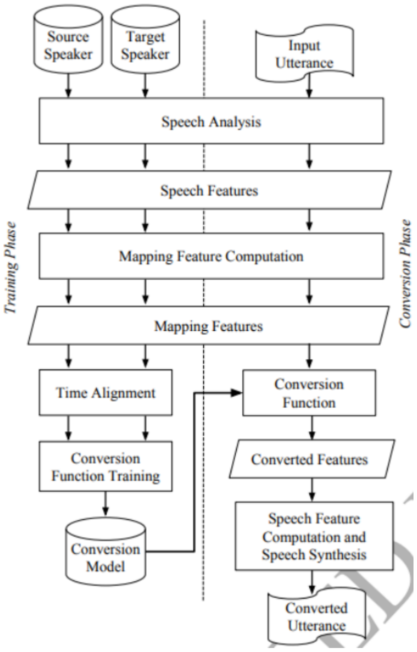

Figure 1. A scheme of VC process [2]

VC basically consist of two phases such as training phase and conversion phase:

- In the training phase it is required to collect data from source and target speakers. Then, collected data is evaluated and its' distinctive features are extracted and formulated by the system. Training phase also can be separated into three main steps:
    - Acoustic analysis acquires some distinctive features of source and target voices in order to define characteristics.
    - Alignment stage detects resembling elements in both source and target speakers.
    - Training step creates a model to convert source to target speakers. Also, we can say that conversion algorithm is involved in this stage.
- The conversion phase utilizes this formulation to resemble source voice to target voice. [1]

II. Explanation of Reference Paper [3]

A. CycleGAN

Purpose of the study is to obtain mapping by train source and target data which are not parallel. CycleGAN is preferred concept as a base network which generates mapping with the benefits of two essential loss term; adversarial loss and cycle-consistency loss.

Adversarial Loss

Adversarial loss defines the difference between converted data and target data. As smaller as loss means the distribution of the converted data is more similar to the target data distribution. Mathematical explanation:

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

First term of the equation measures the ability of generating real target data and next term measures the ability of generating target data considering source data. The mathematical addition of two terms results adversarial loss.

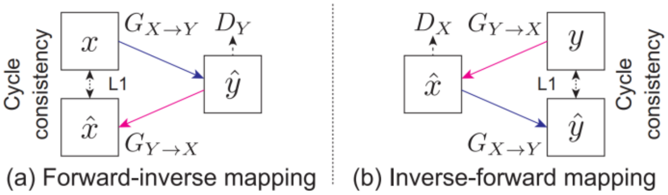

Figure 2. Training procedure of CycleGAN

Cycle-Consistency Loss

Adversarial loss is not adequate for preserve the contextual information of source data. Cycle consistency loss updates the generator models for each iteration considering the difference between generated data and input data.

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

B. CycleGAN for parallel-data-free VC: CycleGAN-VC

Gated CNN and identity mapping loss methods are used in order to apply CycleGan for parallel-data-free VC.

Gated CNN

RNN is claimed to be a better way in terms of the parallel implementations of speech structures which are sequential and hierarchical. For this study gated CNN is an efficient method to realize language and speech modeling.

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

The data-driven activation function is gated linear units (GLUs). Data transmission from layer to layer is elaborately realized considering the previous layer by means of the Gated CNN model.

Identity-Mapping Loss

Cycle-consistency loss is not a sufficient solution for mappings to preserve linguistic-information. Identity-mapping loss provides generator to obtain mapping which preserves the arrangement between input and output.

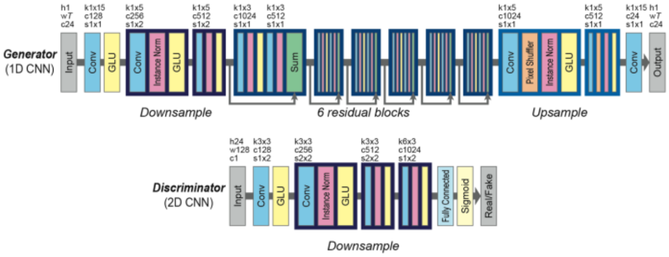

Figure 3. "Network architectures of generator and discriminator. In input or output layer, h, w, and c represent height, width, and number of channels, respectively. In each convolutional layer, k, c, and s denote kernel size, number of channels, and stride size, respectively."

III. Dataset

In the reference project, VC Challenge 2016 dataset was used for training. VCC-16 dataset contains English audio samples recorded by 5 females and 5 males; each speaker has 216 sentences (audio files) which is roughly takes 13 minutes of speech. One-fourth of these samples separated for validation; first half of the remaining samples are assigned for source training and another half for target training for non-parallelism. At the end, they reached 5-6 minutes of duration for training per each speaker.

We have changed the duration of each audio files and the number of these audio files by considering the paper: network does not depend on the input length T because the generator is fully convolutional. [3] Our custom dataset is shared online, link is given in the project web page.

A. Source

Google's text-to-speech voices are used to generate 13 audio clips (each in a duration of approx. 40 secs) in a total of at least 8 minutes for each speaker.

- Female Speaker: *WaveNet Turkish Female voice G*
- Male Speaker: *WaveNet Turkish Male voice E*

B. Target

Similarly, 13 audio clips in a total duration of 8.9 minutes of Turkish news-presenter Ece Uner's speech is chosen.

IV. Implementation & Training

We highly utilized from *Lei Mao's work* [4] while constructing this project. Google's Colaboratory is used to train our model due to lack of high-performance GPUs. Some updates are required to reduce the time-consuming process as a main reason. The training for one model took approximately 5.5 hours after these updates with NVIDIA Tesla T4 GPUs. Two models trained for female-to-female and male-to-female voice conversion. Here is the detailed changelog:

Hyper-parameters

- Training hyper-parameters, audio processing and default path variables are separated from related scripts; gathered together into 'hyparams.py'
- *'decay_threshold'* parameter is added for monitoring the learning rate reduction over iterations.
- In the reference implementation, iteration size dependent on the number of given training audio files not the length; therewithal, learning rate decays with iterations to converge to global minima. However, our dataset and file organization are different and old hyper-parameters result in stop of learning. Therefore, below figure is the plot of new learning rates for both generator and discriminator over growing epochs and iterations.

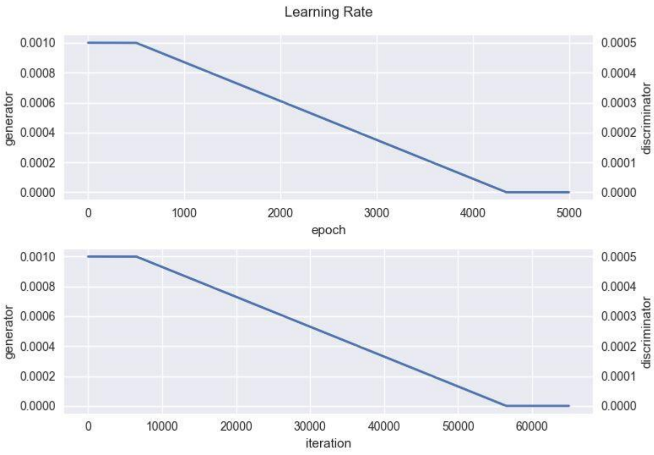

Figure 4. Learning Rates for Generator and Discriminator

Performance

- After realized the training per epoch is so slow because of model-saving and validation operations; *'check_epoch'* parameter is added to control them.
- With the help of another if condition, epoch duration is decreased from approx. 55 seconds to 4 seconds. (NVIDIA Tesla T4)
- Validation functions for conversion from B-to-A is removed. (We only need A-to-B)

Fix

- From now on, epoch range starts from 1 instead of 0.

Miscellenous

- All converted voices generated in validation are stored now; '-CONV-#-EPOCH' extension is added to converted voice filenames to observe the progress.
- Elapsed time per epoch sensitivity is edited in the order-of-milliseconds.
- Never-used scripts and folders are removed.

V. Result

It is not possible to demonstrate audio samples on this paper with sound; thus, synthesized fake voices are uploaded to deployed project web page. The demo allows you to follow the progress on the conversion of input voice over number of epochs trained models. Link can be found in abstract section.

Below figures are obtained from training logs by help of *Tensorboard* framework. For all loss plots in this result section, horizontal axis is iteration number, vertical is the loss value; the orange and blue curves represent female-to-female model and male-to-female model; letter A symbolize the source and B target voice. (65k iterations = 5k epochs)

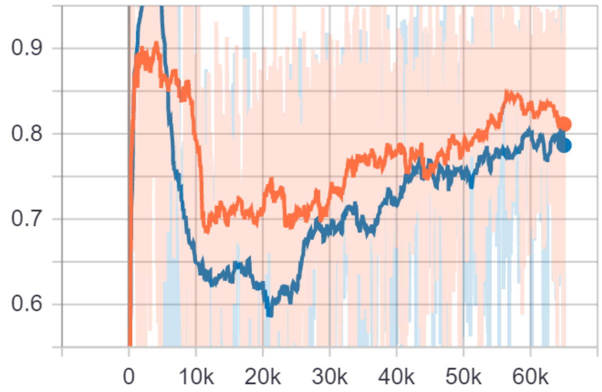

Figure 5. Generator Loss A-to-B

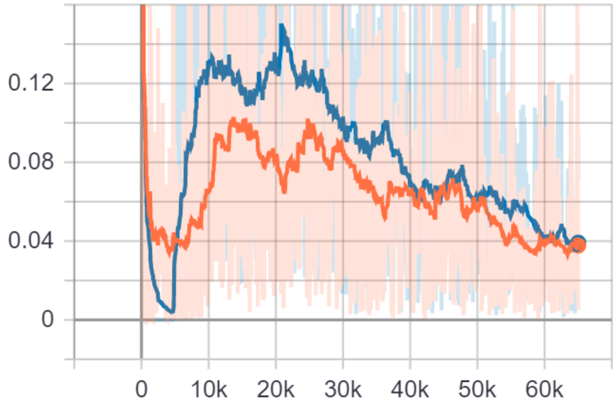

Figure 6. Discriminator Loss B

Figures 5 and 6, show the characteristics of Generative Adversarial Networks that while generator loss decrease, the discriminator loss starts to increase.

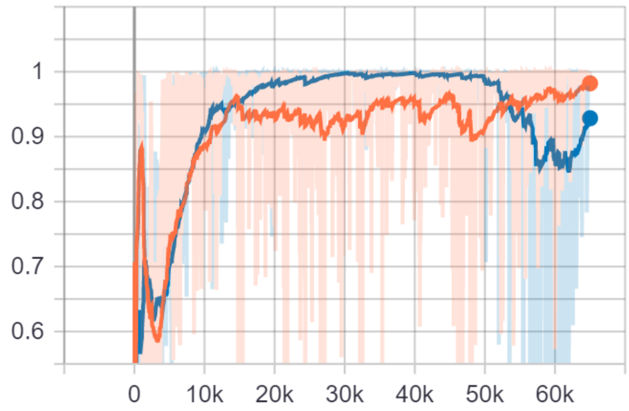

Figure 7: Generator Loss B-to-A

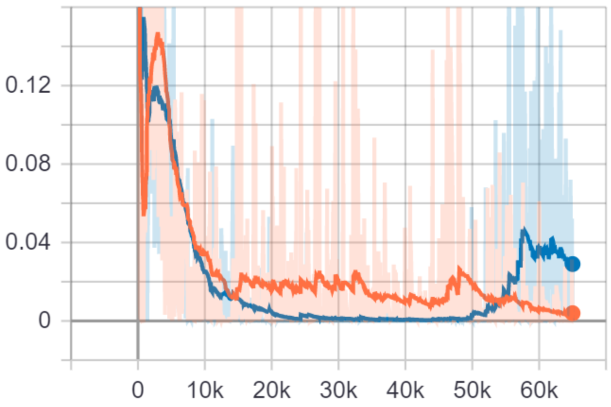

Figure 8: Discriminator Loss A

Same adversarial characteristics can be seen in reverse conversion (see Figure 7 & 8), too. However, if the loss convergence is considered target-to-source conversion seems to give better results. It should be kept in mind that 97% smoothing is applied to the above figures to extract the relationship.

Finally, the next two figures are confirmation of the network converges to our desire by reduction of cycle and generator loss in average case. (Smoothing percentage is 60% by default)

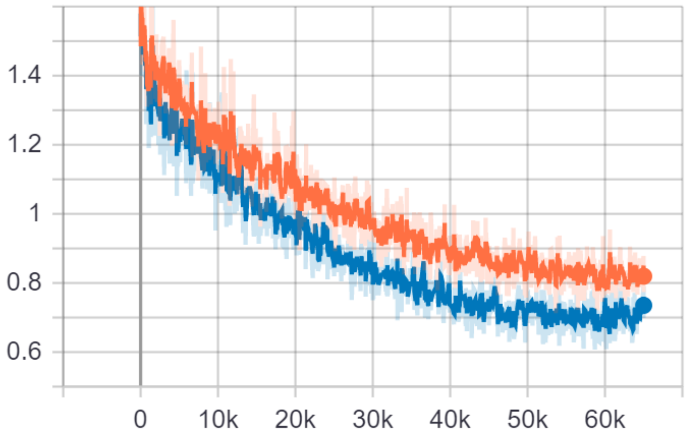

Figure 9: Cycle Loss

References

[1] Ezzine, Frikha. (2017). *A Comparative Study of Voice Conversion Techniques: A review*

[2] Mohammedi, Kain. (2017). *An Overview of Voice Conversion Systems*

[3] T. Kaneko, H. Kameoka. (2017). *Parallel-Data-Free Voice Conversion Using Cycle-Consistent Adversarial Networks*

[4] Lei Mao. *Voice Converter Using CycleGAN and Non-Parallel Data Available at: https://github.com/leimao/Voice-Converter-CycleGAN*

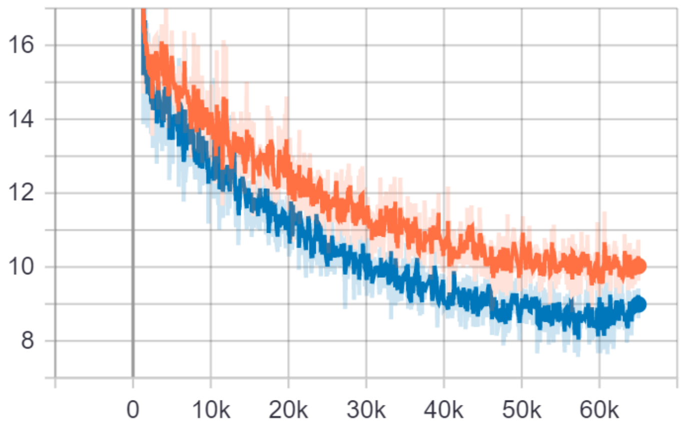

Figure 10: Generator Loss