MELCYCLEGAN - ACCENT CONVERSION FOR FOREIGN SPEAKERS

Liang-yu Chen (lc3533@colmbia.edu)

Columbia University

ABSTRACT

Accent conversion (AC) is a voice conversion (VC) task featuring domain translation on accents aiming to transform audio signals with one accent to another. This can be achieved by building a mapping from one or more accent domains to the target domain. In this paper, we will propose an unsupervised mechanism to convert Indian English audios to native American English audios using deep neural networks. We propose an adversarial generative model MelCycleGAN with relationship preserving Siamese Network based on the previous work [1] and [2] [3].

Index Terms— Accent Conversion, Generative Model, Adversarial Model, Speech Synthesis, Metric Learning, Signal Processing

1. INTRODUCTION

Although the recent development in the natural language processing field has made voice assistants such as Google Home and Alexa more accessible than ever, there have been reported cases of how these voice assistants could discriminate against different accents [4]. According to [4], due to data's existing biases, these voice assistants have less familiarity with some accents and could not recognize some accents at times. These biases in accents have made the voice assistants unresponsive or even isolating to some people. The study of AC addresses the phonetic differences in various accents in the hope of alleviating the existing speech recognition models' biases. By taking the effort to normalize and unify speech data with various accents [5], we hope that the database could be more adaptable and inclusive to different accents.

Even if clean datasets are provided, building the model for accent conversion can be difficult due to possible mismatches of syllables - different people could say the same sentences in different rhythms. The selection of features could also be difficult since the selected embeddings have to preserve the content while maintaining accent features. Deep learning networks are one of the widely adopted solutions for these kind of problems. Generative models are adopted to reproduce converted audio yet may suffer from slow convergence and unstable training.

Our proposed method addresses the technical challenges in accent conversion problems and demonstrates high-quality results. We derive our model from the current state-of-the-art, MelGAN-VC, to perform accent conversion from Indian speakers to English speakers. Compared to traditional GAN solutions, our proposed method shows more realistic results and faster convergence. We will elaborate on the training pipeline, model structure, and loss definition in Section 3 and show our experimental results in Section 4.

2. RELATED WORK

Our proposed method adopts ideas from previous done AC and VC studies. AC has been addressed with various methods. Previous methods [6][7][8][9] focus on directly modify speech features including formants, spectral envelopes, prosody, and articulatory gestures.

Data driven approach was adopted lately. [10] address the problem with Gaussian Mixture Model (GMM) to perform probabilistic articulatory synthesis. [11] proposed deep learning solutions to create a mapping between articulatory gestures into acoustics. [12] further leverage deep autoencoder structure to perform domain transformation on voice data. [13] adopted the similar encoder-decoder structure to embed accented input to accent-free feature, and decode the feature to target accented output. Recently, generative models are also widely adopted to address VC problems. [14] also shows the ability to leverage Linear Predictive Coding (LPC) features with fully-connected GAN to reproduce realistic audio conversions from Indian speakers to American speakers. [15] proposed MelGAN for conditional waveform synthesis to show the capability of GANs to reliably to generate high quality coherent waveforms. [1] further extends this work and adopts Mel spectrogram as audio feature and proposes as a domain translation method MelGAN-VC to perform a voice conversion and reach the current state-of-the-art.

3. METHODOLOGY

3.1. Features

We adopt the feature extraction method discussed in [1] [15]. For jth audio from the dataset, we use a sampling rate sr and sample the audio into τ_j samples and represent it as a vector, w_j. Formally, $w_j \in \mathbb{R}^{\tau_j}$.

We further define a black box function $f(\cdot|F, L, O)$ which maps an vector of sampled signal w into a corresponding mel spectrogram, where F is the number of mel filterbanks, L being the size of the window, and O denotes the overlapping percentage of each window. We set $F = L(1 - O) = H$, where H is a predefined hop size. The embedded mel spectrogram is denoted by $M_j \in \mathbb{R}^{F \times \lceil \frac{\tau_j}{H} \rceil}$. Formally,

$$M_j = f(w_j|H, L, 1 - \frac{H}{L})$$

3.2. Data preprocessing and training pipeline

We construct a set of mel spectrograms $M^{\mathcal{D}}$ for the source dataset $\mathcal{D}^s$ and target dataset $\mathcal{D}^t$

$$M^{\mathcal{D}} = \{f(w|H, L, 1 - \frac{H}{L})|w \in \mathcal{D}\}$$

We adopt cycleGAN to train a pair of accent converters which learn a one-to-one mapping from a specific syllable of one accent to another. In this case, the alignment of audio data does not affect, and we also assume each syllable appears uniformly in the time domain. To preprocess the spectrogram images so that it fits neural networks, we first normalize each spectrogram in terms of their power (db), and iterate all data to split each mel spectrogram into slices in the sequence of time with each slice having a fixed time duration δ^*_D as the minimum among the entire dataset $\mathcal{D}$. Notice based on our assumption, we dispose slices smaller than the predefined input size of our model. Formally, suppose cycleGAN takes in mel spectrogram images $x \in \mathbb{R}^{H \times T \times 1}$ as inputs, any slice $m \in \mathbb{R}^{F \times \delta}$ where $\delta < T$ will be disposed. Therefore, formulation is shown as the following:

$$\delta^*_{\mathcal{D}} = min_{\delta}\{M_{*,\delta}|M \in M^{\mathcal{D}} \wedge \delta > T\}$$

We also define a split function $s(M|\delta)$ such that it maps an input spectorgram image $M \in \mathbb{R}^{F \times N}$ in to $\lfloor \frac{N}{\delta} \rfloor$ images of height F, width δ and one channel, i.e. $x_{\mathcal{D}} \in \mathbb{R}^{F \times \delta \times 1}$ while discarding shorter slices. Let the set of slices for dataset $\mathcal{D}$ to be $K^{\mathcal{D}}$, thus we have

$$K^{\mathcal{D}} = \{s(M|\delta^*_{\mathcal{D}})|M \in M^{\mathcal{D}}\}$$

The visualization of data processing and the entire training pipeline is also given in Fig 1.

3.3. Discriminator

The discriminator is a neural network with three convolutional layers followed by a dense output layer. It takes an input x and outputs a score of how real x looks. Leaky ReLU is selected as the activation function to ease the potential gradient vanishing problem [16]. In addition, the activation of the output dense layer is not used according to [17] for more stable training. Fig 2 shows the structure of discriminators.

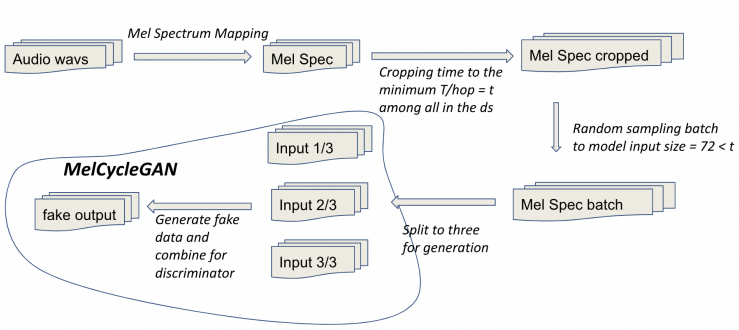

Fig. 1. Training pipeline

We select the hop size $H = 192$ and input size $T = 72$ in our experiment.

3.4. Generator

The generator takes the fully convolutional U-net structure [18] and is composed of three convolutional layers followed by three deconvolutional layers. Leaky ReLU and batch normalization layers are used to deal with gradient vanishing problem.

For the input, we split an input $x \in \mathbb{R}^{H \times T \times 1}$ into q slices in terms of their width, or second dimension so that each slice of image is in the dimension of $\mathbb{R}^{H \times \frac{T}{q} \times 1}$, and the three converted slices are merged back to one image with the original shape as x for the discriminator. Let the function of splitting an input into q sequential slices be $r(x|q)$, so $\sigma = r(x|q)$ and $x' \in \sigma$, $|\sigma| = q$ and $x' \in \mathbb{R}^{H \times \frac{T}{q} \times 1}$. We also define a function of concatenating all $x' \in \sigma$ as $r^{-1}(\sigma|q)$. Note that r^-1 should perform in the order that slices are concatenated in their original order, thus $r^{-1}(r(x|q)|q) = x$. The reason behind is to alleviate the computational requirement of the generator. Furthermore, this measurement helps generators to convert input in a more stable way that correlated inputs are converted into correlated outputs. Fig 3 shows the structure of generators, with $q = 3$.

3.5. Siamese network

A Siamese network [19] is also used to stabilize the training. This network takes an input $x \in \mathbb{R}^{H \times T \times 1}$ and embeds it into a vector of a fixed dimension. We leverage the embedding and calculate the TraVeL loss [3] to speed up the training convergence. We will further discuss the losses in the next subsection. Fig 4 shows the structure of Siamese networks.

3.6. Loss

Our generative model, MelCycleGAN, comprises four major components: G_s generator, G_t generator, D_s discriminator and D_t discriminator. Generators and discriminators take the

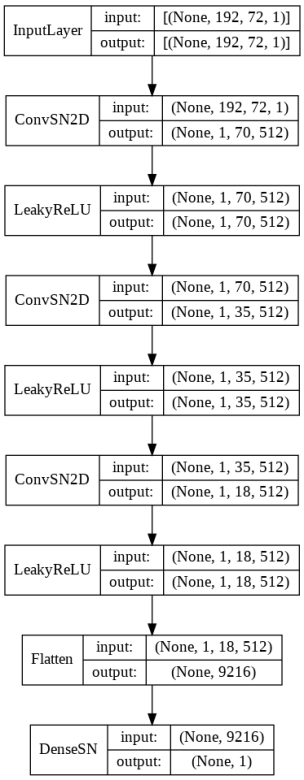

Fig. 2. Structure of discriminators

similar structure as proposed by [1], which uses spectral normalization layers from [20] to enhance the training stability. We define two datasets $\mathcal{D}^s$ and $\mathcal{D}^t$ for accent conversion tasks.

We define the discriminator loss $\mathcal{L}_D$ which penalize discriminators for not predicting low scores on converted spectrogram images and the opposite for real iamges. The discriminator loss for D_s and D_t are formulated as the following:

$$\mathcal{L}_{D_s}(\mathcal{D}^s, \mathcal{D}^t) = E_{\substack{x \sim K^{\mathcal{D}^s} \\ y \sim K^{\mathcal{D}^t}}} \frac{max(0, 1 + D_s(F_s(y))) + max(0, 1 - D_s(x))}{2}$$

where F_s is a converted spectorgram image from using y. Note that the generator takes only a slice of image. So F_s is defined as the fake image generated by G_s that concatenates q converted slices. Formally,

$$F_s(y) = r^{-1}(\{G_s(y')|y' \in r(y|q)\}|q)$$

And the similar definition also applies for D_t:

$$\mathcal{L}_{D_t}(\mathcal{D}^s, \mathcal{D}^t) = E_{\substack{x \sim K^{\mathcal{D}^s} \\ y \sim K^{\mathcal{D}^t}}} \frac{max(0, 1 + D_t(F_t(x))) + max(0, 1 - D_t(y))}{2}$$

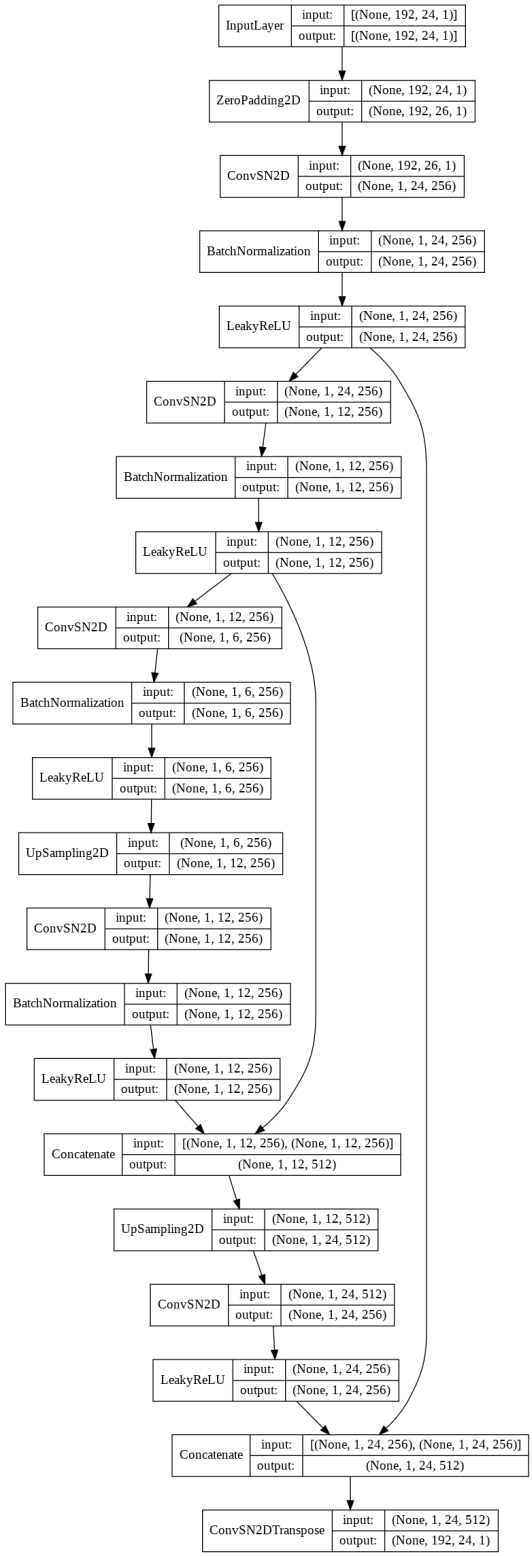

Fig. 3. Structure of generators

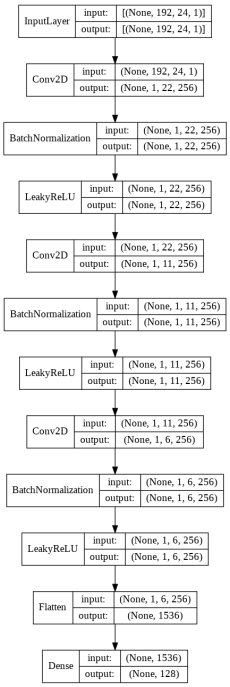

Fig. 4. Structure of Siamese networks

where

$$F_t(x) = r^{-1}(\{G_t(x')|x' \in r(x|q)\}|q)$$

We set the linear combination of adversarial loss $\mathcal{L}_{adv}$, cyclic loss $\mathcal{L}_{cyc}$, identity loss $\mathcal{L}_{id}$ and Siamese loss $\mathcal{L}_{siam}$ to be the total loss $\mathcal{L}_G$ to minimize for the generator. Adversarial loss aims to provide constraints to the converted spectrogram so it can be similar to the target class as possible. It serves as the main constraint to the generator for generating realistic and high resolution audio. Formally, for generator G_s and G_t, the adversarial loss is defined as:

$$\mathcal{L}^s_{adv}(\mathcal{D}^s, \mathcal{D}^t) = -E_{y\sim K^{\mathcal{D}^t}}[D_s(F_s(y))]$$

$$\mathcal{L}^t_{adv}(\mathcal{D}^t, \mathcal{D}^t) = -E_{x\sim K^{\mathcal{D}^s}}[D_t(F_t(x)))]$$

The cyclic loss imposes a constraint on the generator so that the linguistic feature of an input audio still preserves. It is defined as the mean absolute erroer (MAE) between the original slice and the converted slice. In specific,

$$\mathcal{L}^s_{cyc}(\mathcal{D}^t) = E_{y\sim K^{\mathcal{D}^t}}[\frac{1}{q}\sum_{\sigma\in r(y|q)} MAE(\sigma - G_t(G_s(\sigma)))]$$

$$\mathcal{L}^t_{cyc}(\mathcal{D}^s) = E_{x\sim K^{\mathcal{D}^s}}[\frac{1}{q}\sum_{\sigma\in r(x|q)} MAE(\sigma - G_s(G_t(\sigma)))]$$

We also add identity loss to impose a constraint on the generator so that the linguistic information does not lose. Note that cyclic loss and Siamese loss also achieve such purpose, and [2] shows that by adding identity loss to the total loss the training becomes more stable. Identity loss is therefore defined as the following. Note that x is sampled from the same dataset that generator G_s is converting to, and the same applies for G_t.

$$\mathcal{L}^s_{id}(\mathcal{D}^s) = E_{x\sim K^{\mathcal{D}^s}}[\frac{1}{q}\sum_{\sigma\in r(x|q)} MAE(\sigma - (G_s(\sigma)))]$$

$$\mathcal{L}^t_{id}(\mathcal{D}^t) = E_{y\sim K^{\mathcal{D}^t}}[\frac{1}{q}\sum_{\sigma\in r(y|q)} MAE(\sigma - (G_t(\sigma)))]$$

TraVeL loss aims to constrain the Siamese network S to embed each slice of image so that the relationship between the original images are the same as the relationship as the converted images. TraVeL loss for generators is formally defined as:

$$\mathcal{L}^s_{Tra}(\mathcal{D}^t) = E_{y\sim K^{\mathcal{D}^t}}[\frac{1}{|\Omega|}\sum_{\omega\in\Omega}\omega(v_y, v_y^{G_s})]$$

$$\mathcal{L}^t_{Tra}(\mathcal{D}^s) = E_{x\sim K^{\mathcal{D}^s}}[\frac{1}{|\Omega|}\sum_{\omega\in\Omega}\omega(v_x, v_x^{G_t})]$$

where

$$v_\cdot = S(r(\cdot|q)_1) - S(r(\cdot|q)_q)$$
$$v_\cdot^G = S(G(r(\cdot|q)_1)) - S(G(r(\cdot|q)_q))$$

The subscript of $r(\cdot|q)_i$ denotes the ith image of the ordered $r(\cdot|q)$, and Ω is a set of distance metric for two vectors we are adopting such as cosine similarity or Euclidian distance. Note that we only feed the first and last slice of a spectrogram image since middle slices may be highly correlated to its neighboring slices, which opposed to the neural network assumption that input data should be independent and identically distributed. Therefore, the total loss for the generators are denoted as $\mathcal{L}_G$. For G_s and G_t, the losses are:

$$\mathcal{L}_{G_s} = \alpha\mathcal{L}^s_{adv} + \beta\mathcal{L}^s_{cyc} + \gamma\mathcal{L}^s_{id} + \zeta\mathcal{L}^s_{Tra}$$

$$\mathcal{L}_{G_t} = \alpha\mathcal{L}^t_{adv} + \beta\mathcal{L}^t_{cyc} + \gamma\mathcal{L}^t_{id} + \zeta\mathcal{L}^t_{Tra}$$

We set $\alpha = 1$, $\beta = 10$, $\gamma = 0.5$ and $\zeta = 10$ in our experiment setting.

We also define losses to train the Siamese network. Note that by imposing only TraVel loss, the Siamese network may still learn trivial features to minimize the distance, e.g., embed every input into vector of zeros. Thus we add a regularization loss which is a margin-based contrastive objective for Siamese network so that every embedded point is at least δ

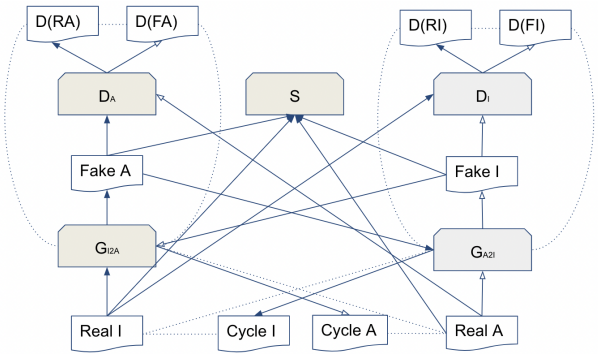

Fig. 5. Structure of MelCycleGAN. "I" denotes data with Indian accent, and "A" denotes data with American accent. Solid line denotes the input/output of the data and dashed line denotes the loss that is calculated. Note that reals and fakes are 1/3 of the input image in terms of time according to our experiment setting.

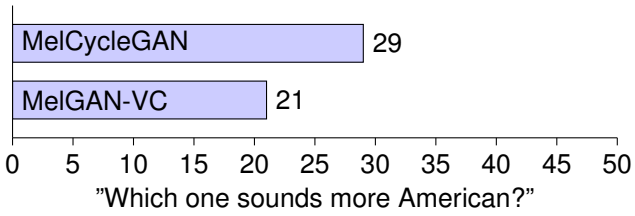

Fig. 6. Realistic accent comparison

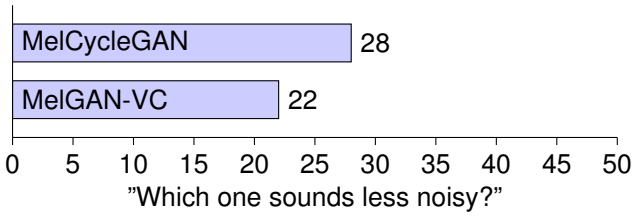

Fig. 7. Quality comparison

way from other points in the embedded space. The Siamese network is used for both G_s and G_t, consequently it takes the loss:

$$\mathcal{L}_{siam}(G_s, G_t, \mathcal{D}^s, \mathcal{D}^t) = \mathcal{L}^s_{Tra} + E_{y\sim K^{\mathcal{D}^t}}[max(0, \delta - ||v_y||_2)] + \mathcal{L}^t_{Tra} + E_{x\sim K^{\mathcal{D}^s}}[max(0, \delta - ||v_x||_2)]$$

The visualization of simplified data flow for MelCycleGAN is also provived in fig 5.

3.7. Audio reconstruction

We apply the inversion of auditory spectrograms that is discussed in [21]. The gradient based method reconstructs the audio based on a sequence of spectorgrams.

4. EXPERIMENT

4.1. Dataset

Two subsets of the CMU Artic dataset [22]: cmu_us_ksp and cmu_us_bdl was used for the experimentation. Both datasets contain 1132 utterances(.wav) spoken by an Indian and an American English male speaker, respectively. Each utterance records a sentence that is usually less than five seconds, with 16KHz waveform. Both speaker are experienced in building synthetic voices. The audio was recorded in a sound-proof room individually.

4.2. Performance

To test the performance of the model and the authenticity of the generated audio, rounds of experiments were primarily conducted. 50 college students with age range from 20 to 30 years old were recruited for the experiment, and each were given a randomly selected converted audio pair, one being the output of MelGAN-VC, and the other being the output of MelCycleGAN.

To measure different dimensions of the authenticity, we asked each participant three questions which correspond to the realisticity, audio quality and linguistic preservation. Fig 6-8 shows the experiment results, where MelCycleGAN out performs MelGAN-VC in every question.

4.3. Speed

We trained MelCycleGAN on a single NVIDIA Tesla P100 GPU on Google Colab Pro. Given a fixed batch size of 64, MelCycleGAN runs 0.31 second per batch, while MelGAN runs 0.033 second per batch. However, MelCycleGAN reaches human identifiable audio after 1 epoch, while it takes 3 epoches to achieve the same quality for MelGAN-VC. MelCycleGAN also converges after 10 epochs, while it requires 25 for MelGAN-VC. Note that by the term converge defined here refers to the fact that human cannot discern the accent difference between the output of the current epoch against the following 5 epochs.

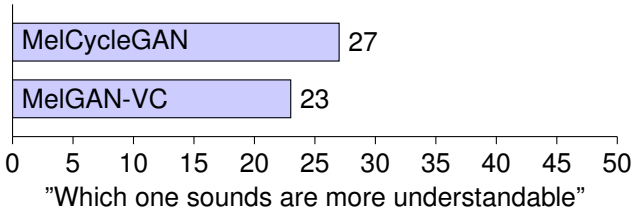

Fig. 8. Linguistic comparison

5. DISCUSSION AND FUTURE WORK

MelCycleGAN gives better results in every aspect in comparison to MelGAN-VC. We believe that by utilizing more GPUs and computational resources MelCycleGAN can achieve the similar runtime per batch as MelGAN-VC with larger memory available. Moreover, MelCycleGAN intrinsically provides two complementary accent converters.

Meanwhile, even though MelCycleGAN and MelGAN-VC both gives realistic results, the runtime becomes a problem when a real-time system is concerned. For a signal of 32k samples under the sample rate of 16k, i.e., a two-second audio, the runtime for the conversion takes around 25 seconds for both methods, where the bottleneck lays in the both the inference of our generator (15 second) and the inversion of spectrogram (9 second) under a single P100. We may need to consider simpler network structure and a quicker reconstruction method in the future to improve the conversion speed.

6. REFERENCES

[1] Marco Pasini, "Melgan-vc: Voice conversion and audio style transfer on arbitrarily long samples using spectrograms," 2019.

[2] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *CVPR 2017*, March 2017.

[3] Matthew Amodio and Smita Krishnaswamy, "Travelgan: Image-to-image translation by transformation vector learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8983–8992.

[4] Drew Harwell, "The accent gap," Available at https://www.washingtonpost.com/graphics/2018/business/alexa-does-not-understand-your-accent/ (2018/07/19).

[5] K Johnson, *Speaker normalization in speech perception*, Blackwell handbooks in linguistics, 2008.

[6] D. Felps, H. Bortfeld, and R. Gutierrez-Osuna, "Foreign accent conversion in computer assisted pronunciation training," *Speech Communication*, vol. 51, no. 10, 2009.

[7] D. Felps, H. Bortfeld, and R. Gutierrez-Osuna, "Foreign accent conversion through concatenative synthesis in the articulatory domain," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 8, 2012.

[8] M. Huckvale and K. Yanagisawa, "Spoken language conversion with accent morphing," *presented at the ISCA Speech Synthesis Workshop,*, 2007.

[9] D. Felps S. Aryal and R. Gutierrez-Osuna, "Foreign accent conversion through voice morphing," *Interspeech*, 2013.

[10] S. Aryal and R. Gutierrez-Osuna, "Accent conversion through cross-speaker articulatory synthesis," *IEEE Transactions on Signal Processing*, vol. 05, 2014.

[11] S. Aryal and R. Gutierrez-Osuna, "Articulatory-based conversion of foreign accents with deep neural networks," *INTERSPEECH 2015*, 2015.

[12] S. H. Mohammadi and A. Kain, "Voice conversion using deep neural networks with speaker-independent pre-training," *2014 IEEE Spoken Language Technology Workshop (SLT)*, 2014.

[13] L. Ai, S. Jeng, and H. Beigi, "Voice conversion using deep neural networks with speaker-independent pre-training," 2020.

[14] Tanmay Chopra, "Lpcgans-ac: A gerative adversarial approach to accent cconversion," *Columba COMS 6998 Speech Recognition*, December 2020.

[15] K.Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. Brebisson, Y. Bengio, and A. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, 2019.

[16] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li, "Empirical evaluation of rectified activations in convolutional network," 2015.

[17] Martin Arjovsky, Soumith Chintala, and Lon Bottou, "Wasserstein gan," 2017.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[19] Luca Bertinetto, Jack Valmadre, Joo F. Henriques, Andrea Vedaldi, and Philip H. S. Torr, "Fully-convolutional siamese networks for object tracking," 2016.

[20] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida, "Spectral normalization for generative adversarial networks," *CoRR*, vol. abs/1802.05957, 2018.

[21] Rémi Decorsière, Peter L. Søndergaard, Ewen N. MacDonald, and Torsten Dau, "Inversion of auditory spectrograms, traditional spectrograms, and other envelope representations," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 23, no. 1, pp. 4656, Jan. 2015.

[22] J. Kominek and A. Black, "The cmu arctic speech databases," *SSW5-2004*, Januar 2004.