WINVC: One-Shot Voice Conversion with Weight Adaptive Instance Normalization

Shengjie Huang[1,2], Mingjie Chen[3], Yanyan Xu[1,2](✉), Dengfeng Ke[4](✉), and Thomas Hain[3]

[1] School of Information Science and Technology, Beijing Forestry University
[2] Engineering Research Center for Forestry-Oriented Intelligent Information Processing of National Forestry and Grassland Administration, Beijing
[3] Computer Science Department, University of Sheffield
[4] School of Information Science, Beijing Language and Culture University
huangshengjie@bjfu.edu.cn, mchen33@sheffield.ac.uk, xuyanyan@bjfu.edu.cn, dengfeng.ke@blcu.edu.cn, t.hain@sheffield.ac.uk

Abstract. This paper proposes a one-shot voice conversion (VC) solution. In many one-shot voice conversion solutions (e.g., Auto-encoder-based VC methods), performances have dramatically been improved due to instance normalization and adaptive instance normalization. However, one-shot voice conversion fluency is still lacking, and the similarity is not good enough. This paper introduces the weight adaptive instance normalization strategy to improve the naturalness and similarity of one-shot voice conversion. Experimental results prove that under the VCTK data set, the MOS score of our proposed model, weight adaptive instance normalization voice conversion (WINVC), reaches 3.97 with five scales, and the SMOS reaches 3.31 with four scales. Besides, WINVC can achieve a MOS score of 3.44 and a SMOS score of 3.11 respectively for one-shot voice conversion under a small data set of 80 speakers with 5 pieces of utterances per person.

Keywords: One-shot voice conversion · Generative adversarial networks (GANs) · Weight adaptive instance normalization.

1 Introduction

Voice conversion aims to preserve the source voice content information while replacing the non-content information in the voice with the target speaker. It has attracted many researchers for its potential applications in security [1], medicine [2], entertainment [3] and education [4].

There are two types of VC, parallel and non-parallel. Due to the difficulty and expensiveness of parallel data collection, several methods based on parallel data, such as the gaussian mixture model (GMM) [5], dynamic time warping (DTW) [6], and deep neural network (DNN) [7], are not particularly effective solutions. In order to overcome this limitation, the phonetic posteriorgrams(PPG)

✉ corresponding authors

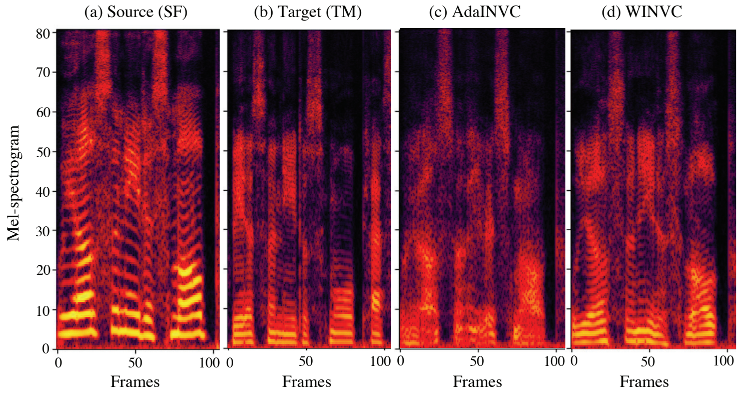

Fig. 1. Comparison of source, target, and converted mel-spectrograms.

based models [8], generative adversarial network (StarGAN) based models [9, 10], and variational auto-encoder (VAE) based models [11] are adopted to solve the problem of non-parallel VC. These methods get rid of the dependence on parallel data. However, when dealing with unseen speakers, a long time adaptation process or a large amount of data is required.

One-shot voice conversion [12–14] and zero-shot voice conversion [15, 16] solve the unseen speaker problem. They convert the source voice to an unseen speaker's voice by referring to only a few target utterances. Moreover, neither the source nor the target utterances appear in the training set during the training phase. They require the model to have a solid ability to separate content information from non-content information in the voice.

Due to the development of the normalization strategy, the performance of the one-shot voice conversion task has been improved. There are two mainstream frameworks for better one-shot VC in recent years, including the auto-encoder based one [12, 15] and the vector quantization (VQ) based one [13, 14]. [15] uses the batch normalization (BN) [17] strategy to implement the one-shot voice conversion successfully. In [13, 14], the instance normalization (IN) [18] strategy is adopted. Compared with the BN strategy used in [15], IN normalizes each input object separately to improve one-shot voice conversion quality. Moreover, AdaINVC [12] innovatively adopts the adaptive instance normalization (AdaIN) [19] strategy. The AdaIN strategy significantly improves the one-shot voice conversion and achieves an improved similarity. Nevertheless, it is challenging to disentangle speaker information and content information through an unsupervised learning method. Moreover, researchers are helpless if the similarity of converted speech is unsatisfying.

In this paper, we propose a weight adaptive instance normalization (WIN) voice conversion system for one-shot VC. The model framework bases on StarGAN-VC2 [10] because it has good effectiveness and convenience, and the model structure is improved. We use the speaker encoder jointly trained with the generator to extract the non-linguistic information of the target speaker. Under the VCTK [20] data set, we compare the WINVC with AdaINVC. The mel-spectrograms

(Fig.1) show that WINVC performs better in content intelligibility and retention, and subjective evaluations show that WINVC achieves better results than AdaINVC on the one-shot voice conversion task. Furthermore, WINVC achieves a competitive one-shot voice conversion performance under the extreme training conditions of using only 80 speakers with 5 utterances per person. In addition, we apply the WIN [21] strategy to AdaINVC, and experimental results show that AdaINVC's one-shot performance has been improved.

To summarize, we list the core contributions of this paper as follows:

1. We design a new model WINVC based on the WIN strategy and StarGAN-VC2. It outperforms the state-of-the-art (SOTA) model AdaINVC naturally and similarly on one-shot voice conversion tasks under non-parallel data.
2. Furthermore, WINVC can perform competitive one-shot voice conversion results even with small amount of data.
3. We also apply the WIN strategy to the previous SOTA model AdaINVC and significantly improves its performance.
4. We use the jointly trained speaker encoder as the non-linguistic information extractor and employ the speaker embedding cycle loss to help the model perform the one-shot VC task better.

2 StarGAN-VC/VC2

This section reviews two previous StarGAN-based voice conversion models: StarGAN-VC [9] and StarGAN-VC2 [10]. As shown in Fig.2, StarGAN-VC uses the StarGAN [22] model for voice conversion, which includes three modules: a generator ($\boldsymbol{G}$), a discriminator ($\boldsymbol{D}$) and a domain classifier ($\boldsymbol{C}$). $\boldsymbol{G}$ takes an acoustic feature sequence $\boldsymbol{x} \in \mathbb{R}$ with an arbitrary attribute and a target attribute label $\boldsymbol{c}$ as the inputs, and generates an acoustic feature sequence,

$$\hat{\boldsymbol{y}} = G(\boldsymbol{x}, \boldsymbol{c}) \tag{1}$$

$\boldsymbol{D}$ is designed to produce a probability $\boldsymbol{D}(\boldsymbol{y}, \boldsymbol{c})$ that an input $\boldsymbol{y}$ is a real speech feature whereas $\boldsymbol{C}$ is designed to produce class probabilities $\boldsymbol{p_C}(\boldsymbol{c} \mid \boldsymbol{y})$ of $\boldsymbol{y}$.

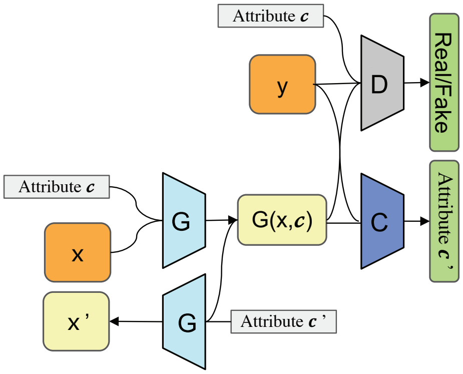

Fig. 2. The architecture of StarGAN-VC.

2.1 Training Objectives

StarGAN-VC/VC2 includes adversarial loss [23], cycle consistency loss [24], and identity mapping loss [25]. StarGAN-VC2 deletes classification loss [26] and updates the BN strategy to the CIN strategy. These loss functions are as follows.

Adversarial loss is

$$\begin{aligned}\mathcal{L}_{\text{adv}}^{D}(D)= & -\mathbb{E}_{c\sim p(c),\mathbf{y}\sim p(\mathbf{y}|c)}[\log D(\mathbf{y},c)] \\ & -\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}),c\sim p(c)}[\log(1-D(G(\mathbf{x},c),c))],\end{aligned} \tag{2}$$

$$\mathcal{L}_{\text{adv}}^{G}(G)=-\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}),c\sim p(c)}[\log D(G(\mathbf{x},c),c)]. \tag{3}$$

Cycle-consistency loss is to preserve the composition in conversion, which is presented as follows:

$$\mathcal{L}_{\text{cyc}}\left(G\right)=\mathbb{E}_{c'\sim p(c),\mathbf{x}\sim p(\mathbf{x}|c'),c\sim p(c)}\left[\|G\left(G(\mathbf{x},c),c'\right)-\mathbf{x}\|\right]. \tag{4}$$

Identity-mapping loss is to facilitate input preservation, which is presented as follows:

$$\mathcal{L}_{\text{id}}(G)=\mathbb{E}_{c'\sim p(c),\mathbf{x}\sim p(\mathbf{x}|c')}\left[\|G\left(\mathbf{x},c'\right)-\mathbf{x}\|\right]. \tag{5}$$

Classification loss is to force the generated data to be similar to the target speaker's, which has been abandoned in StarGAN-VC2:

$$\mathcal{L}_{\text{cls}}^{C}(C)=-\mathbb{E}_{c\sim p(c),\mathbf{y}\sim p(\mathbf{y}|c)}\left[\log p_C(c\mid\mathbf{y})\right], \tag{6}$$

$$\mathcal{L}_{\text{cls}}^{G}(G)=-\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}),c\sim p(c)}\left[\log p_C(c\mid G(\mathbf{x},c))\right]. \tag{7}$$

To summarize, the full objectives of StarGAN-VC to be minimized with respect to $\boldsymbol{G}$, $\boldsymbol{D}$ and $\boldsymbol{C}$ are given as:

$$\mathcal{L}_G(G)=\mathcal{L}_{\text{adv}}^{G}(G)+\lambda_{\text{cls}}\mathcal{L}_{\text{cls}}^{G}(G)+\lambda_{\text{cyc}}\mathcal{L}_{\text{cyc}}(G)+\lambda_{\text{id}}\mathcal{L}_{\text{id}}(G), \tag{8}$$

$$\mathcal{L}_D(D)=\mathcal{L}_{\text{adv}}^{D}(D), \tag{9}$$

$$\mathcal{L}_C(C)=\mathcal{L}_{\text{adv}}^{C}(C). \tag{10}$$

2.2 Generator Architectures

In order to improve voice quality, the StarGAN-VC2 model removes the domain classifier module. StarGAN-VC uses the BN [17] strategy, and StarGAN-VC2 uses the CIN [27] strategy instead.

Given an input batch $x\in R^{BCHW}$, $\mathbf{BN}(x)$ normalizes the mean and standard deviations for the individual feature channel:

$$\mathbf{BN}(x)=\gamma_{single}\left(\frac{x-\mu(x)_{batch}}{\sigma(x)_{batch}}\right)+\beta_{single}, \tag{11}$$

where $\gamma, \beta \in R^C$ are affine parameters learned from data. $\mu(x)$,$\sigma(x) \in R^C$ are the mean and standard deviations, computed across batch size and spatial dimensions independently for each feature channel.

[13] and [14] employ the IN [18] strategy of image style conversion to achieve a better one-shot voice conversion performance.

$$\mathrm{IN}(x) = \gamma_{single}\left(\frac{x - \mu(x)_{sample}}{\sigma(x)_{sample}}\right) + \beta_{single}, \tag{12}$$

where x is the input feature. γ and β form a single set of affine parameters learned from data. μ and σ are computed across spatial dimensions independently for each channel and each sample.

StarGAN-VC2 [10] uses the conditional instance normalization (CIN) [27] strategy, as shown in equation (13), where $\gamma(e_{xy})$ and $\beta(e_{xy})$ are domain-specific scales and bias parameters that allow transforming the modulation in a domain-specific manner. e_{xy} is selected depending on both the source domain code e_x and the target domain code e_y.

$$CIN(x, e_{xy}) = \gamma_{styles}(e_{xy})\left(\frac{x - \mu(x)_{sample}}{\sigma(x)_{sample}}\right) + \beta_{styles}(e_{xy}), \tag{13}$$

$$e_{xy} = \mathrm{concat}([e_x, e_y]). \tag{14}$$

3 The Proposed Model

3.1 Workflow

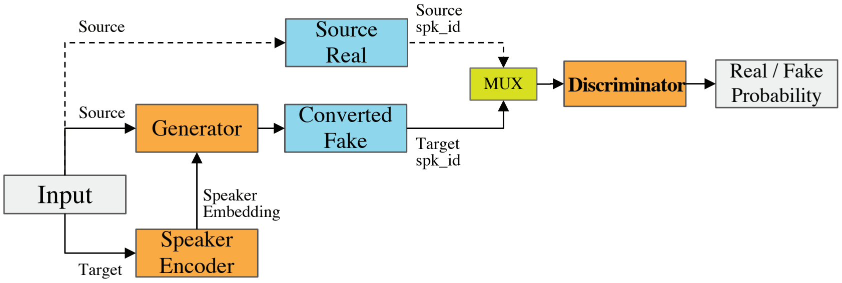

Fig. 3. The workflow diagram of the proposed model

This section introduces the various modules and implementation details of our proposed model[5]. The entire workflow is shown in Fig.3, consisting of a Generator, a Discriminator, and a Speaker Encoder, where **MUX** means randomly sending source real data or converted fake data to Discriminator.

[5] Further details may be found in our implementation code: https://github.com/One-Shot-Voice-Conversion-with-WIN/WINVC

3.2 The generator with weight adaptive instance norm

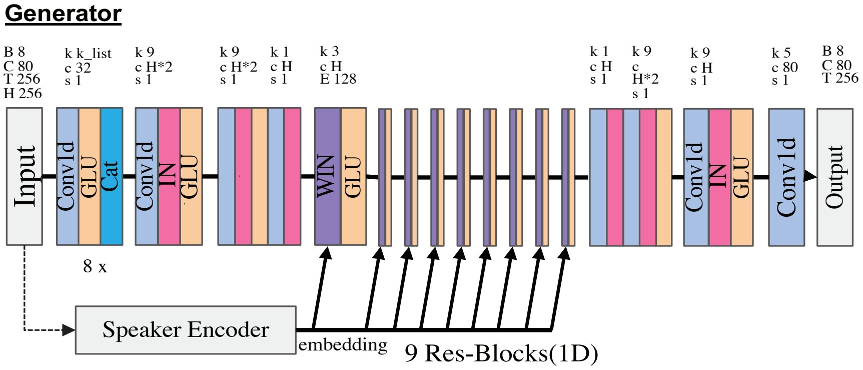

Fig. 4. The module details of G. In input, output, and res-block layers, B, C, T ,H and E represent batch, channel, the number of frames, the hidden size and the embedding size of speaker embedding respectively. In each convolution layer, k, c, and s denote the kernel size, the number of channels and stride, respectively. IN, GLU, Cat and WIN indicate instance normalization, gated linear unit, concatenating and the proposed weight adaptive instance normalization.

As shown in Fig.4, the generator is composed of 1D-convolution, which includes three parts: up-sampling, bottleneck resblocks, and down-sampling. Unlike StarGAN-VC2, our upsampling and downsampling both use the 1D-convolution structure and IN strategy. In the first convolutional layer of upsampling, we use eight different convolution kernel sizes (respectively [1,1,3,3,5,5,7,7]) with 1D-convolution, and finally, concatenate all the 1D-convolution results along the channel dimension. The number of channels of the feature is changed from 80 to a hidden-size of 256. There are nine resblocks in total, all of which composed of WIN modules. The activation function used is gated linear units (GLU).

We propose a new normalization strategy, WIN, into the generator's resblocks. Next, we first introduce AdaIN briefly, and then propose WIN.

Adaptive instance normalization AdaINVC adopts the AdaIN strategy, a particular case of instance normalization, which makes a simple extension to CIN. AdaINVC uses a speaker encoder to extract the speaker embedding $e_y = E(y)$, making it possible to exploit rich information in speaker embedding. The speaker embedding controls the scaling and bias variables of AdaIN. Unlike BN, IN, or CIN, AdaIN has no learnable affine parameters. Instead, it adaptively computes the affine parameters from the style input e_y.

$$\mathrm{AdaIN}(x, e_y) = \sigma(e_y)\left(\frac{x - \mu(x)_{sample}}{\sigma(x)_{sample}}\right) + \mu(e_y). \tag{15}$$

In equation (15), x is a content input to the operator, and e_y is the speaker embedding. $\mu(x)$ and $\sigma(x)$ are the mean and the standard deviations of the feature x across time. $\sigma(e_y)$ and $\mu(e_y)$ are adaptive linear functions. AdaIN (equation 15) performs standard modulation on feature x first, and then uses the adaptive scaling and bias variables, obtained according to the speaker embedding, to perform standard normalization on the features, and finally achieves the integration of feature x and speaker embedding.

Weight adaptive instance normalization To improve the data efficiency of one-shot voice conversion task, we propose the WIN [21]strategy in the bottleneck blocks of the generator, which was initially proposed for image style transfer tasks. Fig.5 illustrates the architectur of WIN[21] module:

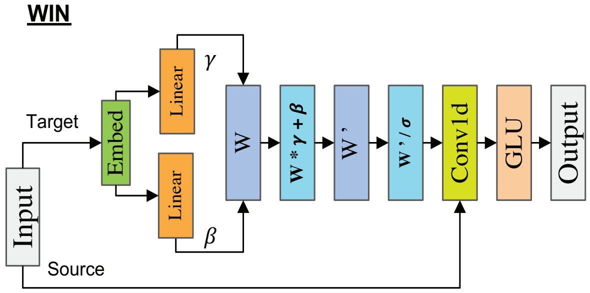

Fig. 5. The architecture of the WIN module.

$$w'_{ijk}(w_{ijk}, e_y) = \gamma_i(e_y) * w_{ijk} + \beta_i(e_y) \tag{16}$$

$$\sigma_j = \sqrt{\sum_{i,k} {w'_{ijk}}^2} \tag{17}$$

$$WIN(w_{ijk}, e_y) = w'_{ijk} / \sqrt{\sum_{i,k} {w'_{ijk}}^2 + \epsilon} \tag{18}$$

In equation (16), w and w' are the original and modulated weights, i denotes the ith input feature map, and j and k enumerate the output feature maps and spatial footprint of the convolution, respectively. e_y is target speaker embedding. In equation (17), σ_j is the standard deviation of modulated weights. In equation (18), ϵ is a small constant to avoid numerical issues.

Different from AdaIN, the demodulation strategy of WIN (equation 18) is related to weight normalization [28]. The modulation (equation 16) and demodulation (equation 18) strategies perform as a part of reparameterizing the weight tensor w. In equation (16), $\gamma_i(e_y)$ and $\beta_i(e_y)$ are two affine transformations applied to speaker embedding e_y corresponding to the ith input feature map, which generate style-dependent scaling and the bias variables. Then they are applied to normalize the convolution weight w_{ijk}, and finally get the intermediate variables w'_{ijk}. In equation (18), we demodulate it again into the convolution weights, which is now embedding related.

The WIN strategy in our proposed WINVC model shows a better one-shot voice conversion performance than the state-of-the-art model AdaINVC. Also, we replace AdaIN with WIN in the baseline AdaINVC. The subjective evaluation shows that WIN enables AdaINVC to achieve a better MOS score and SMOS score, indicating better voice quality and better similarity. Furthermore, the objective evaluation shows that WIN helps AdaINVC get higher speaker verification accuracy.

3.3 The speaker encoder and the discriminator

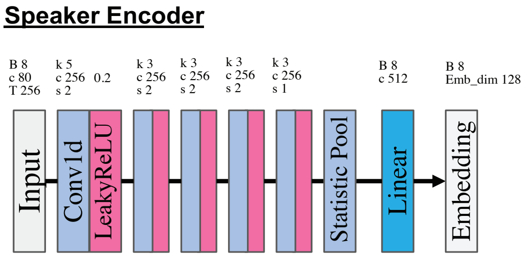

Fig. 6. The architecture of the speaker encoder.

The architecture of the speaker encoder is shown in Fig.6, which adopts a full 1D-convolution form and uses the LeakyRelu activation function after each convolution layer. And it uses a statistic pooling layer as in the xvector [29]. We pass the pool results through a linear function to generate a speaker embedding. Further more, we use a speaker embedding cycle loss (equation 19) to help model get better similarity:

$$\mathcal{L}_{spkcyc} = \cos\left(E\left(x_t\right), E\left(G\left(x_s, E\left(x_t\right)\right)\right)\right), \tag{19}$$

where E is the speaker encoder, x_s and x_t denote the source feature and target feature.

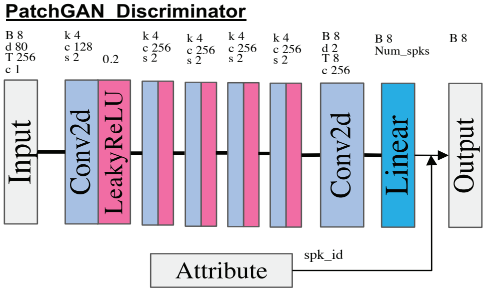

Fig. 7. The module details of D. LeakyRelu indicate LeakyRelu activation. spk_id and Num_spks denote the speaker attribute label and the number of speakers used.

The discriminator structure is shown in Fig.7, which introduces PatchGAN [30] which uses convolution in each layer to reduce parameters and stabilize GAN training. After the last 2D-convolution, the data size obtained is [Batch, num_speakers]. Finally, by specifying the target speaker attribute, the evaluation result of real or fake probability is obtained.

3.4 Training objectives

We use the speaker encoder to extract the non-linguistic information of the target speaker, which is jointly trained with the generator G. And then send the extracted speaker embedding to the generator. G generates the voice conversion result, which is then judged by the discriminator D. In the one-shot stage, AdaINVC, together with most unsupervised models, is helpless if the converted speech's similarity is not satisfactory. However, our model can further improve the similarity of the existing results.

In our proposed model, there are four training objectives: adversarial loss (equation 2,3), cycle consistency loss (equation 4), identity loss (equation 5), and speaker embedding cycle loss (equation 19). The adversarial loss, cycle consistency loss and identity loss are consistent with the corresponding formulas in StarGAN-VC2. The speaker embedding cycle loss is used to calculate the cosine similarity between the converted voice and the ground truth target voice.

Full objective: The full objective is written as

$$\mathcal{L}_D = -\mathcal{L}_{t-adv}, \tag{20}$$

$$\mathcal{L}_G = \lambda_{adv}\mathcal{L}_{t-adv} + \lambda_{spkcyc}\mathcal{L}_{spkcyc} + \lambda_{cyc}\mathcal{L}_{cyc} + \lambda_{id}\mathcal{L}_{id}. \tag{21}$$

where D and G are optimized by minimizing $\mathcal{L}_D$ and $\mathcal{L}_G$ respectively.

4 Experiments

4.1 Datasets

Our experiments are conducted on the VCTK English data set. All selected training utterances are longer than 256 frames. And we use third-party pretrained Parallel WaveGAN [31][6] as vocoder for all comparison models. For the one-shot voice conversion experiment, we use a training dataset of 80 speakers with all utterances, another dataset of 10 unseen speakers, including 5 men and 5 women for unseen-to-unseen one-shot voice conversion. In addition, to further improve the similarity on the existing results, we take an adaption stage, with only one utterance each is used to adapt the pretrained model quickly, and the objective evaluation (Fig.11) show that the similarity can quickly upgrade within 5,000 iterations. For a fair comparison, we make the training set of AdaINVC also contain the 10 unseen speakers with one utterance each. In the end, among the 10 one-shot speakers, we use their other voice data to complete the unseen-to-unseen one-shot voice conversion experiments.

[6] https://github.com/kan-bayashi/ParallelWaveGAN

4.2 Training details

The learning rates of G and D are 2e-4 and 1e-4, respectively. The batch size is 8, and the minimum length of the training data is 256 frames. The values of λ_{id}, λ_{cyc}, λ_{adv} and λ_{spkcyc} are 2, 4, 1 and 5. The WIN convolution kernel size is 3. The number of training iterations is 100k, with the training converging in 10 hours on a single 1080ti. The further adaption stage can be converged within half hour with 5k iterations.

4.3 Subjective evaluations

We analyze the performance[7] differences among the ground truth VCTK utterances (***Target***), our proposed model trained with (80 speakers × all utterances) and adapted with another (10 speakers × 1 utterance)(***WINVC***), the proposed model trained with (80 speakers × 5 utterances) and adapted with another (10 speakers × 1 utterance) (***WINVC5***), the baseline model trained with (80 speakers × all utterances + 10 speakers × 1 utterance) (***AdaINVC***), and the baseline model replaces AdaIN strategy with WIN strategy and is also trained with (80 speakers × all + 10 speakers × 1 utterance) (***AdaINVC_W***).

We conduct mean opinion score (MOS) tests, similarity mean opinion score (SMOS) tests, and ABX tests. The target ground truth utterances (Target) are used as anchor samples. Evaluation utterances are selected based on gender combination for each model. Each gender combination includes 2 pairs of speakers. Each pair of speakers have 20 utterances. Each model is evaluated with $4 \times 2 \times 20 = 160$ utterances. Each utterance is evaluated once. All subjective tests are evaluated with 13 participants.

MOS As shown in Fig.8, ”F” means ”female”, ”M” means ”male”. For example, ”F-M” denotes that female source voice is converted into male target voice, and so on. ”Target” means the ground truth voice of corresponding target speaker. In the subjective naturalness test (MOS), WINVC achieves the highest MOS scores. WINVC5 trained with few data can also achieve a competitive results. In addition, the MOS score of AdaINVC_W is higher than AdaINVC, which indicates that the WIN strategy can indeed make AdaINVC achieve more natural results.

SMOS Fig.9 shows the similarity SMOS. WINVC achieves the highest SMOS scores, WINVC5 can also achieve competitive results. The scores of WINVC and WINVC5 are very close, and all outperform AdaINVC, which denotes that WINVC5 with low resource of training data can also achieve nice similarity. And AdaINVC_W also performs better than AdaINVC, this indicates that the WIN strategy can indeed make AdaINVC achieve better similarity results.

[7] For more details, please refer to the website: https://one-shot-voice-conversion-with-win.github.io

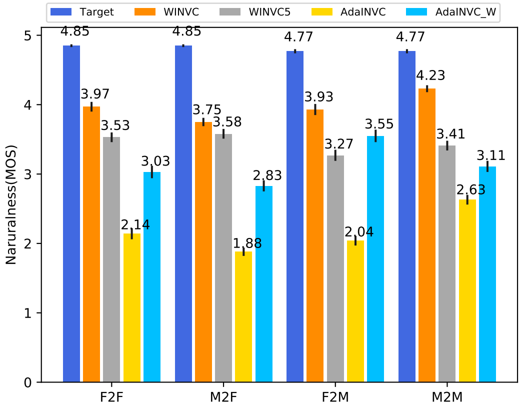

Fig. 8. Naturalness results for baseline model and our proposed WINVC model with 95% confidence intervals.

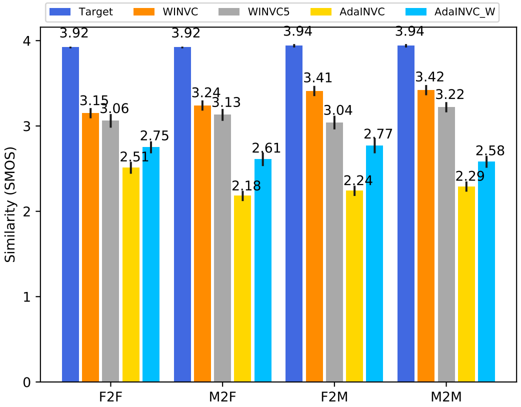

Fig. 9. Naturalness results for baseline model and our proposed WINVC model with 95% confidence intervals.

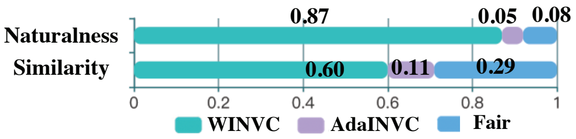

Fig. 10. The ABX test between WINVC and AdaINVC from the aspects of naturalness and similarity.

The ABX test As shown in Fig.10, in the ABX tests, participants need to choose better voice conversion results for the samples of WINVC and AdaINVC from two aspects: similarity and naturalness. From the results, we can conclude that WINVC achieves significant results compared to AdaINVC. Together with the results of MOS and SMOS, which indicate that WIN strategy can indeed enhance the performance AdaINVC, and WINVC can perform better one-shot voice conversion task than AdaINVC from both naturalness and similarity.

4.4 Objective evaluations

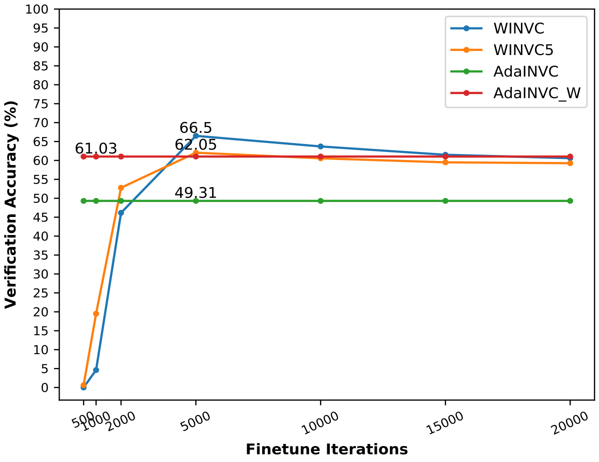

Fig. 11. Comparison of speaker verification accuracy between WINVC and AdaINVC. For better similarity comparision, one unseen utterance of each unseen speaker is used for quick adaption.

The speaker verification accuracy We use speaker verification accuracy as objective metrics. The speaker verification accuracy measures whether the transferred voice belongs to the target speaker. For fair comparison, we used a xvector [29] pretrained with all data of VCTK to verify the speaker identity from the converted voices. As shown in Fig.11, the verification accuracy of our model is obviously higher than that of AdaINVC after quick adaption with 5,000 iterations. Further more, WINVC5 trained with only 5 utterances each speaker, and achieve competitive accuracy as well. After replacing the AdaIN strategy in AdaINVC with the WIN strategy, AdaINVC_W achieved better similarity than AdaINVC.

Disentanglement discussion In addition to the speaker verification accuracy comparison with AdaINVC, we conduct a t-SNE [32] visualization of the latent spaces of the WINVC model. As shown in Fig.12, speaker embeddings from the same speaker are well clustered, and speaker embeddings from different speakers separate in a clean manner. The clear pattern indicates our speaker encoder can verify the speakers' identity from the voice samples.

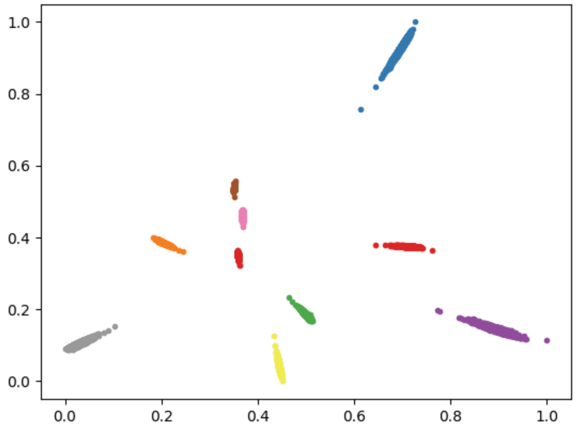

Fig. 12. t-SNE visualization for speaker embeddings of WINVC. The embeddings are extracted from the voice samples of 10 different one-shot speakers. 3,000 embeddings for each person.

5 Conclusions

In this paper, we proposed a novel WIN strategy. In addition, we proposed a WINVC model to perform one-shot voice conversion under the condition of multi-speaker non-parallel data, which achieved significant results. Furthermore, even with a smaller amount of training data, it has achieved a better performance from subjective and objective evaluations than the baseline model, which trained with a larger amount of training data. Besides, with the help of the WIN strategy, the baseline model also performed better. Based on this work, the cross-lingual one-shot voice conversion can be further studied in the future.

Acknowledgement. This work was supported by the Fundamental Research Funds for the Central Universities (grant number 2021ZY87).

References

1. Sisman, B., Zhang, M., Sakti, S., Li, H., Nakamura, S.: Adaptive wavenet vocoder for residual compensation in gan-based voice conversion. In: 2018 IEEE Spoken Language Technology Workshop (SLT). pp. 282–289. IEEE (2018)
2. Nakamura, K., Toda, T., Saruwatari, H., Shikano, K.: Speaking aid system for total laryngectomees using voice conversion of body transmitted artificial speech. proceedings of interspeech sept (2006)
3. Villavicencio, F., Bonada, J.: Applying voice conversion to concatenative singing-voice synthesis. In: Eleventh annual conference of the international speech communication association (2010)
4. Mohammadi, S.H., Kain, A.: An overview of voice conversion systems. Speech Communication **88**, 65–82 (2017)
5. Godoy, E., Rosec, O., Chonavel, T.: Voice conversion using dynamic frequency warping with amplitude scaling, for parallel or nonparallel corpora. IEEE Transactions on Audio, Speech, and Language Processing **20**(4), 1313–1323 (2011)
6. Toda, T., Saruwatari, H., Shikano, K.: High quality voice conversion based on gaussian mixture model with dynamic frequency warping (2001)
7. Nakashika, T., Takashima, R., Takiguchi, T., Ariki, Y.: Voice conversion in high-order eigen space using deep belief nets. In: Interspeech. pp. 369–372 (2013)
8. Sun, L., Li, K., Wang, H., Kang, S., Meng, H.: Phonetic posteriorgrams for many-to-one voice conversion without parallel data training. In: 2016 IEEE International Conference on Multimedia and Expo (ICME). pp. 1–6. IEEE (2016)
9. Kameoka, H., Kaneko, T., Tanaka, K., Hojo, N.: Stargan-vc: Non-parallel many-to-many voice conversion using star generative adversarial networks. In: 2018 IEEE Spoken Language Technology Workshop (SLT). pp. 266–273. IEEE (2018)
10. Kaneko, T., Kameoka, H., Tanaka, K., Hojo, N.: Stargan-vc2: Rethinking conditional methods for stargan-based voice conversion. Proc. Interspeech 2019 pp. 679–683 (2019)
11. Kameoka, H., Kaneko, T., Tanaka, K., Hojo, N.: Acvae-vc: Non-parallel many-to-many voice conversion with auxiliary classifier variational autoencoder. arXiv preprint arXiv:1808.05092 (2018)
12. Chou, J.c., Lee, H.Y.: One-shot voice conversion by separating speaker and content representations with instance normalization. Proc. Interspeech 2019 pp. 664–668 (2019)
13. Wu, D.Y., Lee, H.y.: One-shot voice conversion by vector quantization. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 7734–7738. IEEE (2020)
14. Wu, D.Y., Chen, Y.H., Lee, H.y.: Vqvc+: One-shot voice conversion by vector quantization and u-net architecture. Proc. Interspeech 2020 pp. 4691–4695 (2020)
15. Qian, K., Zhang, Y., Chang, S., Yang, X., Hasegawa-Johnson, M.: Autovc: Zero-shot voice style transfer with only autoencoder loss. In: ICML (2019)
16. Zhang, Z., He, B., Zhang, Z.: Gazev: Gan-based zero-shot voice conversion over non-parallel speech corpus. Proc. Interspeech 2020 pp. 791–795 (2020)
17. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. PMLR (2015)
18. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6924–6932 (2017)

19. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1501–1510 (2017)
20. Veaux, C., Yamagishi, J., MacDonald, K., et al.: Superseded-cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (2016)
21. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8110–8119 (2020)
22. Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8789–8797 (2018)
23. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.C., Bengio, Y.: Generative adversarial nets. In: NIPS (2014)
24. Zhou, T., Krahenbuhl, P., Aubry, M., Huang, Q., Efros, A.A.: Learning dense correspondence via 3d-guided cycle consistency. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 117–126 (2016)
25. Taigman, Y., Polyak, A., Wolf, L.: Unsupervised cross-domain image generation. arXiv preprint arXiv:1611.02200 (2016)
26. Odena, A., Olah, C., Shlens, J.: Conditional image synthesis with auxiliary classifier gans. In: International conference on machine learning. pp. 2642–2651. PMLR (2017)
27. Dumoulin, V., Shlens, J., Kudlur, M.: A learned representation for artistic style. arXiv preprint arXiv:1610.07629 (2016)
28. Salimans, T., Kingma, D.P.: Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In: NIPS (2016)
29. Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., Khudanpur, S.: X-vectors: Robust dnn embeddings for speaker recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 5329–5333. IEEE (2018)
30. Li, C., Wand, M.: Precomputed real-time texture synthesis with markovian generative adversarial networks. In: European conference on computer vision. pp. 702–716. Springer (2016)
31. Yamamoto, R., Song, E., Kim, J.M.: Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 6199–6203. IEEE (2020)
32. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)