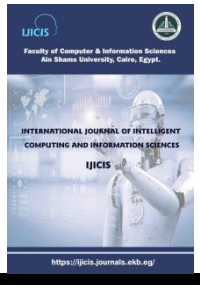
IJICIS, Vol.25, No.2, 42-53
DOI: 10.21608/ijicis.2025.388149.1396

International Journal of Intelligent Computing and Information Sciences

https://ijicis.journals.ekb.eg/



VOICE CONVERSION USING PERTURBATION AUTOVC AND ADAPTIVE INSTANCE NORMALIZATION

Yasmin Alaa*
Computer Science Department, Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt
yasmin.AlaaEldin@cis.asu.edu.eg

Mostafa M. Aref
Computer Science Department, Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt
mostafa.aref@cis.asu.edu.eg

Marco Alfonse
Computer Science Department, Faculty of Computer and Information Sciences, Ain Shams University, Cairo, Egypt
marco_alfonse@cis.asu.edu.eg

Received 2025-05-22; Revised 2025-06-18; Accepted 2025-06-30

Abstract: *Perturbation AutoVC is a model that is derived from the AutoVC model. AutoVC is an autoencoder model that performs Voice Conversion (VC) using non-parallel data and self-reconstruction loss only to train the model. Although AutoVC is a simple and easy model, it has a problem of the bottleneck layer which is used to separate the linguistic content from the speaker information. To tackle this bottleneck problem, Perturbation AutoVC appeared to remove the need of the bottleneck layer to achieve the VC task. In this paper, we use the Perturbation AutoVC as it achieves promising results while changing the way the speaker information is conditioned through using a normalization technique called Adaptive Instance Normalization (AdaIN) instead of the channel-wise concatenation. We setup two experiments seen-to-seen (many-to-many) VC and zero-shot (any-to-any) VC to compare our proposed model with Perturbation AutoVC. We use VCTK corpus (training and testing) and LibriTTS dataset (testing). In seen-to-seen, our proposed model and Perturbation AutoVC achieve d-vector cosine similarity of 0.65 and 0.64 respectively, Mean-Opinion-Score (MOSNet) of 3.48 and 3.32 respectively, Character Error Rate (CER) of 0.09 and 0.13 respectively and Word Error Rate (WER) of 0.15 and 0.22 respectively. In Zero-shot (any-to-any), our proposed model and Perturbation AutoVC achieve MOSNet of 3.40 and 3.16 respectively, d-vector cosine similarity of 0.60 and 0.59 respectively, CER of 0.065 and 0.073 respectively and WER of 0.11 and 0.12 respectively.*

Keywords: *Voice conversion, Any-to-any voice conversion, AutoVC, Adaptive instance normalization, Perturbation AutoVC, Zero-shot voice conversion*

1. Introduction

***Corresponding Author**: Yasmin Alaa

Computer Science Department, Faculty of Computer and Information Science, Ain Shams University, Cairo, Egypt

Email address: yasmin.AlaaEldin@cis.asu.edu.eg

Voice Conversion (VC) refers to the process of modifying the speaker identity in a speech signal from a source speaker to a target speaker while maintaining the original linguistic content. This involves altering speaker-dependent features while preserving speaker-independent information [1]. VC has a wide range of applications, including privacy protection, personalized text-to-speech systems, customized synthetic voices for individuals with vocal impairments, and movie dubbing [1, 2].

There are various versions of the voice conversion (VC) task, which are primarily distinguished based on two fundamental aspects: the type of training data used and how many source and target speakers involved in the conversion process. Regarding the training data, it can be broadly categorized into two main types: parallel data and non-parallel data. Parallel speech data comprises recordings where different speakers articulate the same or substantially equivalent textual content. This type of data facilitates more straightforward alignment between the two voices but requires meticulous data preparation and is often expensive and labor-intensive to collect. Unlike parallel data, non-parallel data doesn't need identical sentences or phrases spoken by the source and target speakers. As a result, non-parallel data is more flexible, easier to obtain, and significantly more practical for large-scale or real-world applications. Due to these advantages, a significant portion of ongoing research in VC field have focused on developing models that are capable of functioning effectively with non-parallel data. Besides the training data, voice conversion systems can also be classified by how many source and target speakers they support. These categories include one-to-one VC, many-to-many VC, and zero-shot VC, which is also referred to as any-to-any VC. In one-to-one voice conversion, the goal is to change the voice of a particular source speaker to sound like a particular target speaker. Although this setup is simple to implement, its applicability is quite restricted. On the other hand, many-to-many VC involves training models to perform conversions among a group of multiple speakers, allowing for a more dynamic and versatile system capable of dealing with a wide variety of speaker combinations encountered during training. The most advanced and flexible setup is zero-shot or any-to-any VC, in which the system is capable of converting voices between speakers that it has never encountered during training. This form of VC is particularly attractive for real-world deployment, as it allows the system to generalize to entirely new voices without requiring additional retraining or speaker-specific data. Overall, the current trend in VC research places a strong emphasis on non-parallel data as well as many-to-many and zero-shot (any-to-any) conversion tasks. This focus stems from their greater scalability, practicality, and relevance to real-world applications, where data collection resources are often limited and user-specific customization is highly desirable.

To achieve non-parallel many-to-many and non-parallel zero-shot (any-to-any) VC, various VC models that use different technologies have appeared. These technologies include Generative Adversarial Nets (GANs) [3], encoder-decoder [4,5], Automatic Speech Recognition (ASR) [6], Vector Quantization (VQ) [7] and others. StarGAN-VC [8] and StarGAN-VC2 [9] are two of the models that are based on GAN and are considered as extensions to CycleGAN-VC model [10]. CycleGAN-VC model succussed to use the non-parallel data instead of the parallel one by introducing a new loss called cycle consistency loss but it couldn’t achieve the many-to-many task. StarGAN-VC could solve the many-to-many VC task by adding another loss to the CycleGAN-VC model which is called domain classification loss. Although the successful usage of GAN in VC, the models based on GAN suffer from the known training difficulties of GAN. Encoder-decoder architecture achieves VC through removing speaker information from the input speech while preserving the linguistic information using a bottleneck (encoder). Then, this linguistic information along with the target speaker information is processed by the decoder to produce a speech contains the source linguistic content along with the target speaker identity. Some of the models that use this architecture are AutoVC [4] and model in [5]. Encoder-decoder based models often have the over-

smoothing problem and the VC quality relies heavily on choosing the right dimension for the bottleneck layer. ASR can be used in VC task in two directions, one of them is using ASR as feature extractor for the linguistic data presented in the input speech and the other direction is using ASR to encourage the VC model to learn removing the speaker-dependent features from the input speech. Some of the models that use ASR to achieve VC task are presented in [11,12,13]. ASR-based models are dependent on an ASR module which needs to be pretrained using a large dataset of the used language. Also, VQ can be used the same way as ASR to help the model learn preserving the linguistic data while removing the speaker characteristics. One of the models that use VQ is VQMIVC [14]. VQ-based models are dependent on the VQ technique which needs a trainable code book dependent on the used language. Using such pretrained external modules (ASR module and VQ code book) seems to limit the ability of the VC models as they are mainly dependent on the accuracy of these external modules and their availability for the language used in the conversion.

Although AutoVC achieves a better voice conversion than the other VC models using a simple training schema without the need for any external modules and can achieve zero-shot (any-to-any) voice conversion, it has the bottleneck problem. To tackle the bottleneck problem in AutoVC, Perturbation AUTOVC [15] has appeared. Perturbation AutoVC is inspired by neural analysis and synthesis (NANSY) [16] in using perturbation to remove speaker characteristics in the input speech instead of using a bottleneck. In this work, we adapted the same architecture of Perturbation AutoVC by changing the conditional method of the speaker identity to achieve higher similarity and consequently higher speech quality. We used Adaptive Instance Normalization (AdaIN) [17] to condition the speaker identity instead of channel-wise concatenation. Using normalization to condition the speaker identity is inspired by Activation Guidance and Adaptive Instance Normalization VC (AGAIN-VC) [18], model in [19] and StarGAN-VC2 models. This conditional method significantly contributed to the success of AGAIN-VC model and model in [19] in achieving one-shot VC and the success of StarGAN-VC2 model in achieving better speech similarity and quality than the StarGAN-VC model. Concatenation, while simple, can lead to complex interactions between the concatenated features that are hard for the network to untangle. This can result in audible artifacts or a less natural quality in the converted speech. Normalization, by providing a structured way to combine content and style, often yields smoother, more natural-sounding outputs.

The rest of the paper is organized as follows: Section 2 reviews related work, Section 3 presents the architecture of the proposed model, Section 4 describes the experimental setup, Section 5 discusses the results and provides analysis, and finally, Section 6 concludes the paper with a summary of our contributions.

2. Related Works

2.1 AutoVC

AutoVC achieves zero-shot (any-to-any) VC through autoencoder architecture which is trained only using the reconstruction loss. AutoVC's main goal is to remove speaker information from the input speck while keeping linguistic content through a well-designed bottleneck layer. Fig. 1 shows the AutoVC's architecture.

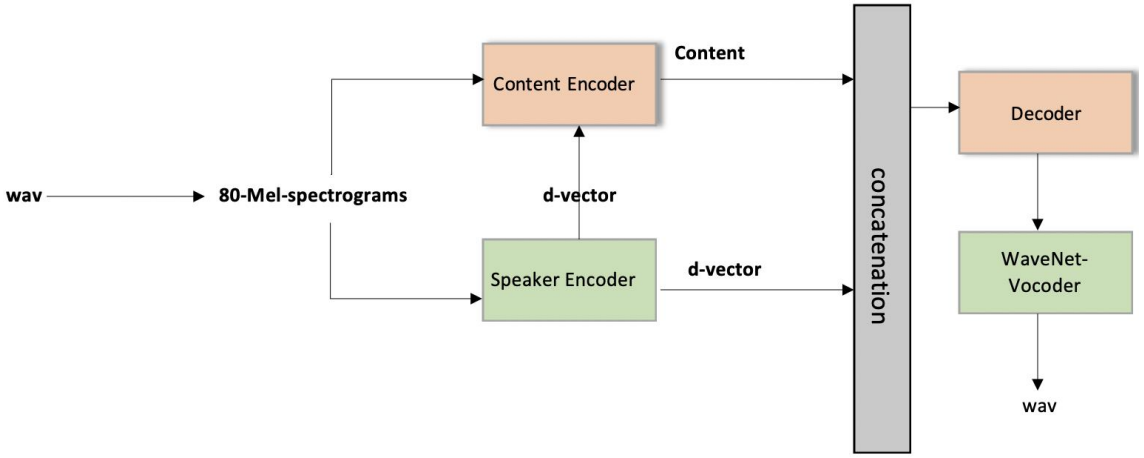


Fig. 1. AutoVC model's architecture.

AutoVC consists of 4 main modules: 1) encoder to disentangle linguistic content from the input speech, 2) decoder to generate the target speech which contains source linguistic content along with the target speaker identity, 3) speaker-encoder to generate speaker embedding, and 4) vocoder to generate the waveform. The encoder's input is the 80-mel-spectrogram of the source speech along with the source identity to help the encoder disentangle linguistic content from speaker identity. The speaker-encoder is a pre-trained network [20], its input is the 80-mel-spectrogram to produce speaker identity representation (d-vector). The decoder's input is the latent codes produced by the encoder (representation for content information) and the target speaker characteristics (d-vector). Vocoder is a pre-trained network by the authors using the method in [21] responsible for turning the converted 80-mel-spectrograms into the waveform. Although AutoVC achieves zero-shot (any-to-any) VC using a simple training schema, its quality is heavily dependent on the dimension of the content-encoder's bottleneck layer. This bottleneck dimension needs to be carefully determined (it can't be neither narrow nor wide) which is hard, needs many trails and creates a trade-off between speech quality (preserving source linguistic content) and target speaker similarity.

2.2 Perturbation AutoVC

Perturbation AutoVC comes to solve the above-mentioned bottleneck problem in AutoVC by depending on a perturbation process instead of the bottleneck architecture to remove speaker information and preserve the linguistic information inspired by NANSY. Perturbation [22, 23] means to distort unneeded information in the input speech while preserving the needed ones which is linguistic information in our case. Following NANSY perturbation AutoVC achieves the perturbation process by applying three functions random frequency shaping using a parametric equalizer (peq), pitch randomization (pr) and formant shifting (fs). Fig. 2 shows Perturbation AutoVC's architecture.

Perturbation AutoVC consists of 4 main components: Perturbation, Speaker Encoder, Content Encoder and Decoder. Perturbation component is responsible for applying the above mentioned three functions to the input waveform as follows fs(pr(peq(X))) where X is the input waveform. The perturbation process output after it is converted to 80-dimensional mel-spectrogram goes through the content encoder. Content Encoder produces linguistic content embedding while ignoring the distorted components of the input mel-spectrograms. In the Perturbation AutoVC, there is no bottleneck in the content-encoder as the bottleneck dimension (down-sampling factor) of the original model (AutoVC) is set to 1. Speaker-encoder, a pre-

trained network by the authors, extracts x-vector from the input mel-spectrograms which represents the speaker identity. Decoder is responsible for taking the content embedding, x-vector (speaker characteristics) and energy (extracted from the original input mel-spectrograms) to generate the converted mel-spectrograms. Energy is determined by taking the average of the log mel-spectrograms across the frequency axis. Using Energy as an additional input to the decoder is inspired by NANSY. To generate converted waveform HiFi-GAN [24] is used.

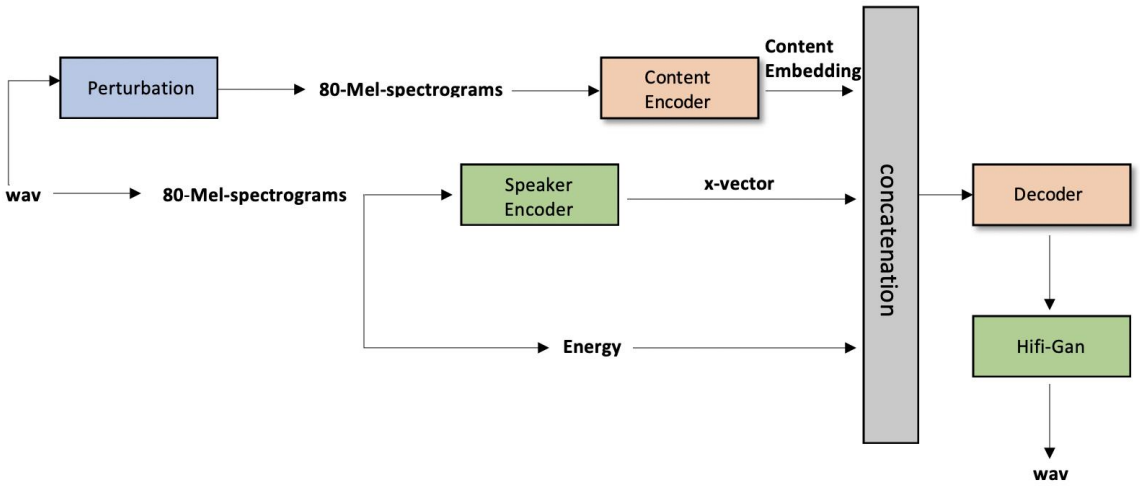


Fig. 2. The architecture of the Perturbation AutoVC model.

2.3 Adaptive Instance Normalization (AdaIN)

AdaIN is a normalization technique designed to align the channel-wise mean and variance of content features with those of style features. AdaIN is calculated as in Eq. 1.

$$AdaIN(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y) \quad (1)$$

Where y is the style and x is the content.

AdaIN can be used in conditioning any content with a desired style. We inspired by the Star-GanVC2 to use normalization as a conditional method for the speaker identity which was one of the main reasons that made Star-GanVC2 achieves higher quality and similarity. Inspired by AGAIN-VC model and model in [19], we used AdaIN to have the ability of achieving zero-shot (any-to-any) conversion while using normalization as our conditional method.

3. Proposed Model

Our proposed model as shown in Fig.3 contains of 5 main modules: 1) content-encoder 2) decoder 3) speaker-encoder 4) perturbation module 5) AdaIN.

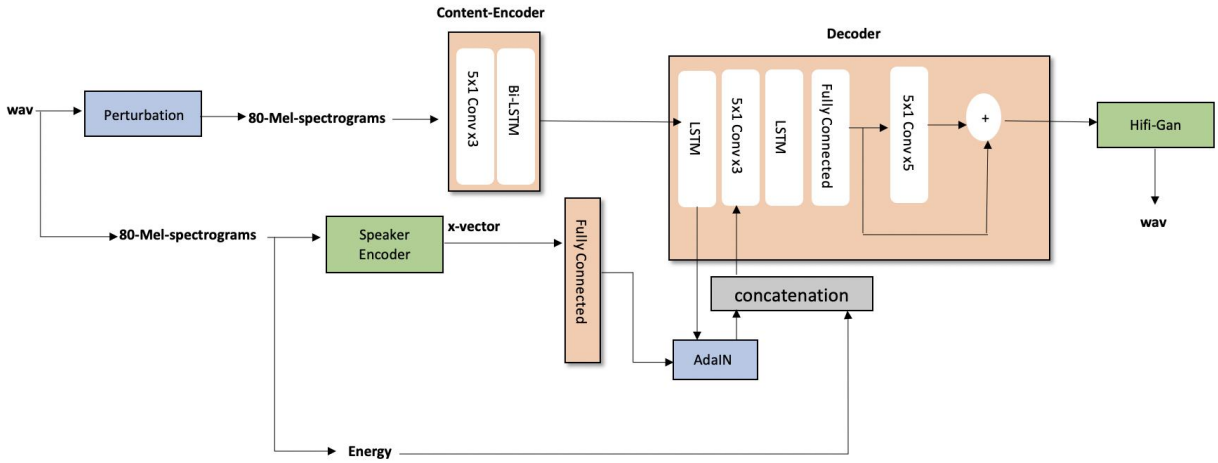


Fig. 3. Our proposed model architecture.

Following the Perturbation AutoVC, we apply perturbation to the input waveform through a sequence of the above mentioned three functions. Perturbated waveform equals fs(pr(peq(X))) where X is the input wav, then from this perturbated waveform 80-mel-spectrograms are extracted then go through the content-encoder.

Content-encoder has the same architecture as the Perturbation AutoVC, it takes the 80-perturbated-mel-spectrogrmas through a 3 5x1 convolutional layers then the output goes through a bi-directional LSTM to produce 512-content-embeddings.

For speaker-encoder, we use the pretrained model by the Perturbation AutoVC ‘s authors that takes 80-mel-spectrograms to produce 192-speaker-embeddings (x-vectors).

Decoder is a model that takes the content-embedding, speaker-embedding and energy (extracted from the 80-mel-spectrograms) to produce the converted 80-mel-spectrograms. Our decoder differs from the Perturbation AutoVC’s decoder in the conditional method. Instead of the channel-wise concatenation to the content and speaker embeddings and the energy, the content-embedding goes through a LSTM then this LSTM’s output and the speaker-embedding go through AdaIN layer then this AdaIN’s result is concatenated with the energy. Since the LSTM output dimension (512) is not the same as the speaker-embedding dimension (192), before AdaIN the speaker-embedding is linearly transformed first to match the LSTM output dimension (512) then Eq. 1 is applied. The output from the AdaIN goes through 3 5x1 convolution layers then the convolutional output goes through an LSTM. Then the LSTM output is fed into a fully connected layer to produce the initial converted 80-mel-spectrograms $\hat{X}_1$. $\hat{X}_1$ goes through 5 5x1 convolutional layers (as a Res-Net) to produce the residual 80-mel-spectrograms $\hat{X}_2$. So, the final output of the decoder is the addition of $\hat{X}_1$ and $\hat{X}_2$ as shown in Eq.2.

$$\hat{X}_{1\to1} = \hat{X}_1 + \hat{X}_2 \quad (2)$$

Where $\hat{X}_{1\to1}$ is the final converted 80-mel-spectrograms produced by the decoder.

Hifi-GAN, as employed in Perturbation AutoVC, is utilized to generate the converted waveform from the 80-mel-spectrograms produced by the decoder.

Our model is dependent only on self-reconstruction loss as AutoVC and Perturbation AutoVC. 80-mel-spectrograms reconstruction loss is computed between the original and initial converted mel-spectrograms as in Eq. 3. Also, 80-mel-spectrogram reconstruction loss is computed between the original and the final converted mel-spectrograms as in Eq. 4. Content-embedding reconstruction is also calculated using Eq. 5. The final training loss is calculated as in Eq. 6.

$$L_{recon0} = E[\| \hat{X}_1 - X_1 \|_2^2] \quad (3)$$

Where X_1 is the original 8-mel-spectrograms.

$$L_{recon} = E[\| \hat{X}_{1\to1} - X_1 \|_2^2] \quad (4)$$

$$L_{content} = E[\| E_c(\hat{X}_{1\to1}) - E_c(X_1) \|_1] \quad (5)$$

Where E_c is the content-encoder.

$$min_{E_C(.),D(.,.)} L = \mu(L_{recon} + L_{recon0}) + \lambda L_{content} \quad (6)$$

Where D is the decoder, μ is a weight for the 80-mel-spectrograms reconstruction loss and λ is a weight for the content-embedding reconstruction loss.

4. Experiments

4.1 Dataset

Our proposed model and the Perturbation AutoVC model are trained on VCTK [25] corpus. VCTK corpus contains records from 109 English speakers, each one has around 400 recorded sentences and all records are down-sampled to 48kHz. For the training part, we used 80% of the speakers and we used all their records except 10 utterances from each speaker are kept for the testing part. Also, LibriTTS [26] (train-clean-100 set) which is derived from LibriSpeech [27] is used in zero-shot (any-to-any) voice conversion testing. We made the same training setup as Perturbation AutoVC. Mel-spectrogram (dimension of 80) is extracted from the training records which are down-sampled to 22.05 kHz. A Fast Fourier Transform (FFT), with a size of 1024, a window size of 1024, and a hop length of 256, is used for the extraction of mel-spectrograms.128 frames from each record are used through truncating or padding the extracted mel-spectrograms. The model is trained using a batch size of 2, a learning rate of 0.0001, and the ADAM optimizer. A value of 1 for λ and a value of 2 for μ are used. We trained the two models for about 80k iterations (reaching reconstruction error of about 0.2).

4.2 Evaluation Methods

Four evaluation methods are used which are d-vector cosine similarity [28], Mean Opinion Score using a pretrained Network (MOSNet) [29,30], Word Error Rate (WER) [31,32] and Character Error Rate (CER) [31,32].

Following [28], we assess the similarity between generated and real speech using d-vector cosine similarity. This metric compares the d-vector extracted from a real recording to that from a generated recording for the same speaker. A higher d-vector cosine similarity value indicates greater resemblance between the generated and real speech. D-vector cosine similarity is calculated as shown in Eq. 7.

$$Cosine\ Similarity(y, \hat{y}) = \frac{y.\hat{y}}{\|y\| \|\hat{y}\|} \quad (7)$$

Where y and yˆ are d-vectors of the target speaker from any one of his/her original speeches and from his/her generated speech respectively and n is the d-vector length.

The dot product between the two d-vectors is calculated as in Eq. 8.

$$y.\hat{y} = \sum_{i=0}^{n=256} y_i \hat{y}_i \quad (8)$$

The magnitude of the d-vector is calculated as in Eq. 9.

$$\| y \| = \sqrt{y.y} \quad (9)$$

Inspired by [30], we used MOSNet [29] which is a pre-trained network that simulates the human opinion on the quality of the generated records. A higher MOSNet value indicates a higher quality of the generated records.

Inspired by [31], we employed CER and WER [32] to quantify the linguistic preservation of the generated recordings. These metrics measure the error rate by calculating the ratio of the sum of deletions, substitutions, and insertions required to transform the candidate sequence into the reference sequence, to the total number of words or characters in the reference. Consequently, a lower value for CER or WER indicates a higher quality of linguistic preservation in the generated recording. WER/CER is calculated as shown in Eq. 10.

$$WER/CER = (S+D+I)/N \quad (10)$$

Where S, D, and I denote the counts of substitutions, deletions, and insertions, respectively, while N refers to the total number of actual words or characters.

4.3 Experimental Setup

We made two experiments in which our Proposed model and Perturbation AutoVC are evaluated using the four mentioned evaluation methods. First experiment is seen-to-seen in which we selected 10 males and 10 females from the seen speakers and used for each one 10 records that are kept for the testing (not showing in the training phase). We made all the possible conversions resulting on 4000 records generated from each model. Second experiment is zero-shot (any-to-any) in which we selected 5 males and 5 females from the train-clean-100 set of LibriTTS dataset and used for each one 5 records. We made all the possible conversions resulting on 500 records generated from each model.

5 Results

5.1 Seen-to-seen Voice Conversion Test

The average of the WER, CER, MOSNet and d-vector similarity values that are calculated from the 4000 generated records are showed in Table. 1.

Table. 1 WER, CER, MOSNet and d-vector cosine similarity for the Proposed and Perturbation AutoVC models.

	WER	CER	MOSNet	d-vector cosine similarity
Our proposed model	**0.15**	**0.09**	**3.48**	**0.65**
Perturbation AutoVC	0.22	0.13	3.32	0.64

In this experiment, our proposed model achieves lower CER and WER, also our proposed model achieves higher MOSNet. For the d-vector cosine similarity, our proposed model and Perturbation AutoVC achieves comparable results.

5.2 Zero-shot (any-to-any) Voice Conversion Test

The average of the WER, CER, MOSNet and d-vector similarity values that are calculated from the 500 generated records are showed in Table. 2.

Table. 2 WER, CER, MOSNet and d-vector cosine similarity for the Proposed and Perturbation AutoVC models.

	WER	CER	MOSNet	d-vector cosine similarity
Our proposed model	**0.11**	**0.065**	**3.40**	**0.60**
Perturbation AutoVC	0.12	0.073	3.16	0.59

In this experiment, our proposed model achieves lower CER, also our proposed model achieves higher MOSNet. For the WER and d-vector cosine similarity, our proposed model and Perturbation AutoVC achieves comparable results.

Based on these comparisons, as well as the findings from our previous work [33] comparing AdaIN-based VC model with other VC models. AdaIN shows its efficiency while being used in Perturbation AutoVC. This claims that using AdaIN as a conditional method instead of channel-wise concatenation in VC models is a good practice to achieve better VC quality.

Generated samples can be found in [34].

6 Conclusion

This work proposes a model which performs zero-shot (any-to-any) non-parallel Voice Conversion (VC). The proposed model leverages the Perturbation AutoVC model with changing the conditional method of the speaker information to be done using a normalization technique called Adaptive Instance Normalization (AdaIN) instead of the channel-wise concatenation. As concatenation, while simple, can lead to complex interactions between the concatenated features that are hard for the network to untangle. This can result in audible artifacts or a less natural quality in the converted speech. Normalization, by providing a structured way to combine content and style, often yields smoother, more natural-sounding outputs. Despite the AdaIN's ability to enable flexible, real-time, and high-quality VC, the quality and duration of the reference utterance for the target speaker significantly impact the effectiveness of AdaIN as noisy or very short reference audios can lead to suboptimal style transfer. Our proposed model depends on the reconstruction loss only. The VCTK Corpus was used for both training and testing, with additional testing conducted on the LibriTTS dataset. We performed two experiments zero-shot (any-to-any) VC and seen-to-seen VC to compare between our proposed model and Perturbation AutoVC. For the

evaluation, we used four metrices: d-vector cosine similarity, Mean-Opinion-Score using pretrained Network (MOSNet), Character Error Rate (CER) and Word Error Rate (WER). Experiments showed that our proposed model achieves better results which indicates that using AdaIN as the conditional method, which is a little modification, can have a good effect in VC models.

References

1. Mohammadi, S. H., & Kain, A. (2017). An overview of voice conversion systems. Speech Communication, 88, 65-82. https://doi.org/10.1016/j.specom.2017.01.008
2. Sisman, B., Yamagishi, J., King, S., & Li, H. (2020). An Overview of Voice Conversion and Its Challenges: From Statistical Modeling to Deep Learning. IEEE/ACM Trans. Audio, Speech and Lang. Proc., 29, 132–157. https://doi.org/10.1109/TASLP.2020.3038524
3. Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative adversarial nets. Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, 2672–2680. Presented at the Montreal, Canada. Cambridge, MA, USA: MIT Press. https://dl.acm.org/doi/10.1145/3690624.3709392
4. Qian, K., Zhang, Y., Chang, S., Yang, X., & Hasegawa-Johnson, M. (2019). AutoVC: Zero-shot voice style transfer with only autoencoder loss. International Conference on Machine Learning(ICML). Proceedings of Machine Learning Research, 97, 5210–5219.
5. Luong, M., & Tran, V.-A. (08 2021). Many-to-Many Voice Conversion Based Feature Disentanglement Using Variational Autoencoder. Interspeech, 851–855. doi:10.21437/Interspeech.2021-2086
6. Chan, W., Jaitly, N., Le, Q., & Vinyals, O. (03 2016). Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. ICASSP, 4960–4964. doi:10.1109/ICASSP.2016.7472621
7. van Niekerk, B., Nortje, L., & Kamper, H. (10 2020). Vector-Quantized Neural Networks for Acoustic Unit Discovery in the ZeroSpeech 2020 Challenge. Interspeech, 4836–4840. doi:10.21437/Interspeech.2020-1693
8. Kameoka, H., Kaneko, T., Tanaka, K., & Hojo, N. (2018). StarGAN-VC: non-parallel many-to-many Voice Conversion Using Star Generative Adversarial Networks. 2018 IEEE Spoken Language Technology Workshop (SLT), 266–273. doi:10.1109/SLT.2018.8639535
9. Kaneko, T., Kameoka, H., Tanaka, K., & Hojo, N. (09 2019). StarGAN-VC2: Rethinking Conditional Methods for StarGAN-Based Voice Conversion. Interspeech, 679–683. doi:10.21437/Interspeech.2019-2236
10. Kaneko, T., & Kameoka, H. (2018). CycleGAN-VC: Non-parallel Voice Conversion Using Cycle-Consistent Adversarial Networks. 2018 26th European Signal Processing Conference (EUSIPCO), 2100–2104. doi:10.23919/EUSIPCO.2018.8553236
11. Biadsy, F., Weiss, R., Moreno, P., Kanevsky, D., & Jia, Y. (09 2019). Parrotron: An End-to-End Speech-to-Speech Conversion Model and its Applications to Hearing-Impaired Speech and Speech Separation. Interspeech, 4115–4119. doi:10.21437/Interspeech.2019-1789
12. Sun, L., Li, K., Wang, H., Kang, S., & Meng, H. (2016). Phonetic posteriorgrams for many-to-one voice conversion without parallel data training. 2016 IEEE International Conference on Multimedia and Expo (ICME), 1–6. doi:10.1109/ICME.2016.7552917
13. Zheng, W.-Z., Han, J.-Y., Lee, C.-K., Lin, Y.-Y., Chang, S.-H., & Lai, Y.-H. (2022). Phonetic posteriorgram-based voice conversion system to improve speech intelligibility of dysarthric

patients. Computer Methods and Programs in Biomedicine, 215, 106602. doi:10.1016/j.cmpb.2021.106602
14. Wang, D., Deng, L., Yeung, Y., Chen, X., Liu, X., & Meng, H. (08 2021). VQMIVC: Vector Quantization and Mutual Information-Based Unsupervised Speech Representation Disentanglement for One-Shot Voice Conversion. Interspeech, 1344–1348. doi:10.21437/Interspeech.2021-283
15. Park, H.-Y., Lee, Y. H., & Chun, C. (2023). Perturbation AUTOVC: Voice Conversion From Perturbation and Autoencoder Loss. IEEE Access, 11, 140174–140185. doi:10.1109/ACCESS.2023.3341434
16. Choi, H.-S., Lee, J., Kim, W., Lee, J. H., Heo, H., & Lee, K. (2021). Neural analysis and synthesis: reconstructing speech from self-supervised representations. Proceedings of the 35th International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc, 34, 16251–16265. doi:10.5555/3540261.3541504
17. Huang, X., & Belongie, S. (2017). Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization. 2017 IEEE International Conference on Computer Vision (ICCV), 1510–1519. doi:10.1109/ICCV.2017.167
18. Chen, Y.-H., Wu, D.-Y., Wu, T.-H., & Lee, H.-Y. (2021). Again-VC: A One-Shot Voice Conversion Using Activation Guidance and Adaptive Instance Normalization. ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 5954–5958. doi:10.1109/ICASSP39728.2021.9414257
19. Chou, J.-C., Yeh, C.-C., & Lee, H.-Y. (2019). One-shot Voice Conversion by Separating Speaker and Content Representations with Instance Normalization. Interspeech, 664-668. doi:10.21437/Interspeech.2019-2663
20. Wan, L., Wang, Q., Papir, A., & Moreno, I. L. (2018). Generalized End-to-End Loss for Speaker Verification. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 4879–4883. doi:10.1109/ICASSP.2018.8462665
21. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K. (2016) WaveNet: A Generative Model for Raw Audio. Proc. 9th ISCA Workshop on Speech Synthesis Workshop (SSW 9), 125. doi:10.48550/arXiv.1609.03499
22. Jaitly, N., & Hinton, G. E. (2013, June). Vocal tract length perturbation (VTLP) improves speech recognition. In Proc. ICML workshop on deep learning for audio, speech and language, 117, 21. Retrieved from https://api.semanticscholar.org/CorpusID:14140670
23. Ning, Z., Xie, Q., Zhu, P., Wang, Z., Xue, L., Yao, J., … Bi, M. (2023). Expressive-VC: Highly Expressive Voice Conversion with Attention Fusion of Bottleneck and Perturbation Features. ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 1–5. doi:10.1109/ICASSP49357.2023.10096057
24. Kong, J., Kim, J., & Bae, J. (2020). HiFi-GAN: generative adversarial networks for efficient and high fidelity speech synthesis. Proceedings of the 34th International Conference on Neural Information Processing Systems, 33,17022-17033. Presented at the Vancouver, BC, Canada. Red Hook, NY, USA: Curran Associates Inc. doi:abs/10.5555/3495724.3497152
25. Veaux, C., Yamagishi, J., & MacDonald, K. (2017). *SUPERSEDED - CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit*, University of Edinburgh, The Centre for Speech Technology Research (CSTR), Accessed on: March 12, 2023. [Online]. Available: https://doi.org/10.7488/ds/1994
26. Zen, H., Dang, V., Clark, R., Zhang, Y., Weiss, R. J., Jia, Y., … Wu, Y. (2019). LibriTTS: A corpus derived from LibriSpeech for text-to-speech. in Proc. Interspeech, 1526–1530. doi:abs/1904.02882

27. Panayotov, V., Chen, G., Povey, D., & Khudanpur, S. (2015). Librispeech: An ASR corpus based on public domain audio books. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 5206–5210. doi:10.1109/ICASSP.2015.7178964
28. Wang, C., & Yu, Y. (2021). Non-Parallel Many-To-Many Voice Conversion Using Local Linguistic Tokens. ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 5929–5933. doi:10.1109/ICASSP39728.2021.9413540
29. Lo, C.-C., Fu, S.-W., Huang, W.-C., Wang, X., Yamagishi, J., Tsao, Y., & Wang, H. (2019). MosNet: Deep learning based objective assessment for voice conversion. Interspeech, 1541-1545. doi:10.21437/Interspeech.2019-2003
30. Eskimez, S.E., Dimitriadis, D., Kumatani, K., Gmyr, R. (2021). One-shot Voice Conversion with Speaker-agnostic StarGAN. Interspeech, 1334–1338. doi: 10.21437/Interspeech.2021-221
31. Liu, S., Cao, Y., Wang, D., Wu, X., Liu, X., & Meng, H. (2021). Any-to-Many Voice Conversion With Location-Relative Sequence-to-Sequence Modeling. IEEE/ACM Trans. Audio, Speech and Lang. Proc., 29, 1717–1728. doi:10.1109/TASLP.2021.3076867
32. WER Implementation: https://github.com/jitsi/jiwer, Accessed on: January 22, 2024.
33. Alaa, Y., Aref, M. M., & Alfonse, M. (2025). Zero-shot non-parallel voice conversion using auto-encoder and adaptive instance normalization. *Signal, Image and Video Processing*, *19*(8). doi:10.1007/s11760-025-04236-y
34. Converted Samples: https://github.com/yasminalaa/Zero-Shot-Non-Parallel-Voice-Conversion-using-Auto-Encoder-and-Adaptive-Instance-Normalization.git.