AUTOVC: Zero-Shot Voice Style Transfer with Only Autoencoder Loss

Kaizhi Qian [*1] **Yang Zhang** [*2 3] **Shiyu Chang** [2 3] **Xuesong Yang** [1] **Mark Hasegawa-Johnson** [1]

Abstract

Non-parallel many-to-many voice conversion, as well as zero-shot voice conversion, remain under-explored areas. Deep style transfer algorithms, such as generative adversarial networks (GAN) and conditional variational autoencoder (CVAE), are being applied as new solutions in this field. However, GAN training is sophisticated and difficult, and there is no strong evidence that its generated speech is of good perceptual quality. On the other hands, CVAE training is simple but does not come with the distribution-matching property as in GAN. In this paper, we propose a new style transfer scheme that involves only an autoencoder with a carefully designed bottleneck. We formally show that this scheme can achieve distribution-matching style transfer by training only on a self-reconstruction loss. Based on this scheme, we proposed AUTOVC, which achieves state-of-the-art results in many-to-many voice conversion with non-parallel data, and which is the first to perform zero-shot voice conversion.

1. Introduction

The idea of speaking in someone else's voice never fails to be a fascinating element in action and fiction movies, and it also finds its way to many practical applications, *e.g.* privacy and identity protection, creative industry *etc.* In the speech research community, this task is referred to as the voice conversion problem, which involves modifying a given speech from a source speaker to match its vocal qualities with a target speaker.

Despite the continuing research efforts in voice conversion, three problems remain under-explored. First, most voice conversion systems assume the availability of parallel training data, *i.e.* speech pairs where the two speakers utter the same sentences. Only a few can be trained on non-parallel data. Second, among the few existing algorithms that work on non-parallel data, even fewer can work for many-to-many conversion, *i.e.* converting from multiple source speakers to multiple target speakers. Last but not least, no voice conversion systems are able to perform zero-shot conversion, *i.e.* conversion to the voice of an unseen speaker by looking at only a few of his/her utterances.

With the recent advances in deep style transfer, the traditional voice conversion problem is being recast as a style transfer problem, where the vocal qualities can be regarded as styles, and speakers as domains. There are many style transfer algorithms that do not require parallel data, and are applicable to multiple domains, so they are readily available as new solutions to voice conversion. In particular, generative adversarial network (GAN) (Goodfellow et al., 2014) and conditional variational autoencoder (CVAE) (Kingma & Welling, 2013; Kingma et al., 2014), are gaining popularity in voice conversion.

However, neither of GAN and CVAE is perfect. GAN comes with a nice theoretical justification that the generated data would match the distribution of the true data, and has achieved state-of-the-art results, particularly in computer vision. However, it is widely acknowledged that GAN is very hard to train, and its convergence property is fragile. Also, although there is an increasing number of works that introduce GAN to speech generation (Donahue et al., 2018) and speech domain transfer (Pascual et al., 2017; Subakan & Smaragdis, 2018; Fan et al., 2018; Hosseini-Asl et al., 2018), there is no strong evidence that the generated speech ***sounds*** real. Speech that is able to fool the discriminators has yet to fool human ears. On the other hand, CVAE is easier to train. All it needs to do is to perform self-reconstruction and maximize a variational lower bound of the output probability. The intuition is to infer a hypothetical style-independent hidden variable, which is then combined with the new style information to generate the style-transferred output. However, CVAE alone does not guarantee distribution matching, and often suffers from over-smoothing of the conversion output (Kameoka et al., 2018b).

Due to the lack of a suitable style transfer algorithm, existing voice conversion systems have yet to produce satisfactory

[*]Equal contribution [1]University of Illinois at Urbana-Champaign, IL, USA [2]MIT-IBM Watson AI Lab, Cambridge, MA, USA [3]IBM Research, Cambridge, MA, USA. Correspondence to: Kaizhi Qian <kqian3@illinois.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

results, which naturally leads to the following question. Is there a style transfer algorithm that is also theoretically proven to match the distribution as GAN is, and that trains as easily as CVAE, and that works better for speech?

Motivated by this, in this paper, we propose a new style transfer scheme, which involves only a ***vanilla*** autoencoder with a carefully designed bottleneck. Similar to CVAE, the proposed scheme only needs to be trained on the self-reconstruction loss, but it has a distribution matching property similar to GAN's. This is because the correctly-designed bottleneck will learn to remove the style information from the source and get the style-independent code, which is the goal of CVAE, but which the training scheme of CVAE is unable to guarantee.

Based on this scheme, we propose AUTOVC, a many-to-many voice style transfer algorithm without parallel data. AUTOVC follows the autoencoder framework and is trained only on autoencoder loss, but it introduces carefully-tuned dimension reduction and temporal downsampling to constrain the information flow. As we will show, this simple scheme leads to a significant performance gain. AUTOVC achieves superior performance on traditional many-to-many conversion task, where all the speakers are seen in the training set. Also, equipped the speaker embedding trained for speaker verification (Heigold et al., 2016; Wan et al., 2018), AUTOVC is among the first to perform zero-shot voice conversion with decent performance. Considering the quality of the results and the simplicity of its training scheme, AUTOVC opens a new path towards a simpler and better voice conversion and general style transfer systems. The implementation will become publicly available.

2. Related Works

There are several works that perform non-parallel many-to-many voice conversion using VAE and its combination with adversarial training. VAE-VC (Hsu et al., 2016) is a simple voice conversion system using VAE. Afterward, many research focuses on removing the style information from the VAE code. VAW-GAN (Hsu et al., 2017) introduces a GAN on the VAE output. CDVAE-VC (Huang et al., 2018) introduces two VAEs on two spectral features and forced the latent codes of the two features to contain similar information. ACVAE-VC (Kameoka et al., 2018a) introduces an auxiliary classifier on the output to encourage the conversion results to be correctly classified as the target speaker's utterances. Chou et al. (2018) introduce a classifier on the code and a GAN on the output. Similarly, StarGAN (Kaneko & Kameoka, 2017) and CycleGAN (Zhu et al., 2017), which consist of encoder-decoder architectures with GAN, are applied to voice conversion (Kameoka et al., 2018b; Fang et al., 2018). GAN alone is also applied to voice conversion (Gao et al., 2018). However, the conversion quality of these algorithms is still limited. Text transcriptions are introduced to assist the learning of the latent code (Xie et al., 2016; Saito et al., 2018; Biadsy et al., 2019), but we will focus on voice conversion without text transcriptions, which is more flexible for low-resourced languages.

Atalla et al. (2019); Chou et al. (2018); Nachmani & Wolf (2019) conduct research on style transfer using autoencoder, but none has unveiled its distribution-matching property by properly designing the bottleneck.

3. Style Transfer Autoencoder

In this section, we will discuss how and why autoencoder can match the data distribution as GAN does. Although our intended application is voice conversion, the discussion in this section is applicable to other style transfer applications as well. As general mathematical notations, upper-case letters, *e.g.* X, denote random variables/vectors; lower-case letters, *e.g.* x, denote deterministic values or instances of random variables; $X(1:T)$ denotes a random process, with $(1:T)$ denoting a collection of time indices running from 1 to T. For notational ease, sometimes the time indices are omitted to represent the collection of the random process at all times. $p_X(\cdot|Y)$ denotes the probability mass function (PMF) or probability density function (PDF) of X conditional on Y; $p_X(\cdot|Y=y)$, or sometimes $p_X(\cdot|y)$ without causing confusions, denotes the PMF/PDF of X conditional on Y taking a specific value y; similarly, $\mathbb{E}[X|Y]$, $\mathbb{E}[X|Y=y]$ and $\mathbb{E}[X|y]$ denote the corresponding conditional expectations. It is worth mentioning that $\mathbb{E}[X|Y]$ is still a random, but $\mathbb{E}[X|Y=y]$ or $\mathbb{E}[X|y]$. $H(\cdot)$ denotes the entropy, and $H(\cdot|\cdot)$ denotes the conditional entropy.

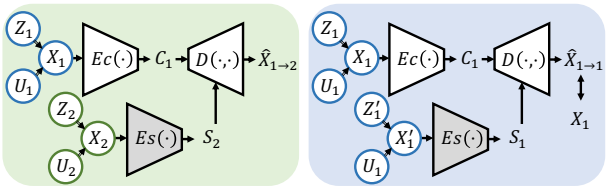
3.1. Problem Formulation

Assume that speech is generated by the following stochastic process. First, a speaker identity U is a random variable drawn from the speaker population $p_U(\cdot)$. Then, a content vector $Z = Z(1:T)$ is a random process drawn from the joint content distribution $p_Z(\cdot)$. Here content refers to the phonetic and prosodic information. Finally, given the speaker identity and content, the speech segment $X = X(1:T)$ is a random process randomly sampled from the speech distribution, *i.e.* $p_X(\cdot|U,Z)$, which characterizes the distribution of the speaker U's speech uttering the content Z. $X(t)$ can represents a sample of speech waveform, or a frame of speech spectrogram. In this paper, we will be working on speech spectrogram. Here, we assume that each speaker produces the same amount of gross information, *i.e.*

$$H(X|U=u) = h_{\text{speech}} = \text{constant}, \tag{1}$$

regardless of u.

Now, assume two sets of variables, (U_1, Z_1, X_1) and (U_2, Z_2, X_2), are independent and identically distributed



(a) Conversion (b) Training

Figure 1. The style transfer autoencoder framework. The ovals denote the probabilistic graphical model of the speech generation process. The grey boxes denote pre-trained modules. (a) During conversion, the source speech is fed to the content encoder. An utterance of the target speaker is fed to the speaker encoder. The decoder produces the conversion results. (b) During training, the source speech is fed to the content encoder. Another utterance of the same *source* speaker is fed to the speaker encoder. The content encoder and the decoder minimize the self-reconstruction error.

(i.i.d.) random samples generated from this process. (U_1, Z_1, X_1) belong to the *source speaker* and (U_2, Z_2, X_2) belong to the *target speaker*. Our goal is to design a speech converter that produces the conversion output, $\hat{X}_{1\to2}$, which preserves the content in X_1, but matches the speaker characteritics of speaker U_2. Formally, an ideal speech converter should have the following desirable property:

$$p_{\hat{X}_{1\to2}}(\cdot|U_2 = u_2, Z_1 = z_1) = p_X(\cdot|U = u_2, Z = z_1). \quad (2)$$

Eq. (2) means that given the target speaker's identity $U_2 = u_2$ and the content in the source speech $Z_1 = z_1$, the converted speech should sound like u_2 uttering z_1.

When U_1 and U_2 are both seen in the training set, the problem is a standard multi-speaker conversion problem, which has been addressed by some existing works. When U_1 or U_2 is not included in the training set, the problem becomes the more challenging zero-shot voice conversion problem, which is also a target task of the proposed AUTOVC.

3.2. The Autoencoder Framework

AUTOVC solves the voice conversion problem with a very simple autoencoder framework, as shown in Fig. 1. The framework consists of three modules, a content encoder $E_c(\cdot)$ that produces a content embedding from speech, a speaker encoder $E_s(\cdot)$ that produces a speaker embedding from speech, and a decoder $D(\cdot,\cdot)$ that produce speech from content and speaker embeddings. The inputs to these modules are different for conversion and training.

Conversion: As shown in Fig. 1(a), during the actual conversion, the source speech X_1 is fed into the content encoder to have content information extracted. The target speech is fed into the speaker encoder to provide target speaker information. The decoder produces the converted speech based on the content information in the source speech and the speaker information in the target speech.

$$C_1 = E_c(X_1), \quad S_2 = E_s(X_2), \quad \hat{X}_{1\to2} = D(C_1, S_2). \quad (3)$$

Here C_1 and $\hat{X}_{1\to2}$ are both random processes. S_2 is simply a random vector.

Training: Throughout the paper, we will assume the speaker encoder is already pre-trained to extract some form of speaker dependent embedding, so by training we refer to the training of the content encoder and the decoder. As shown in Fig. 1(b), since we do not assume the availability of parallel data, only self-reconstruction is needed for training. More specifically, the input to the content encoder is still X_1, but the input to the style encoder becomes an utterance from the same speaker U_1, denoted as X_1'.[1] Then for each input speech X_1, AUTOVC learns to reconstruct itself:

$$C_1 = E_c(X_1), \quad S_1 = E_s(X_1'), \quad \hat{X}_{1\to1} = D(C_1, S_1). \quad (4)$$

The loss the function to minimize is simply the weighted combination of the self-reconstruction error and the content code reconstruction error, *i.e.*

$$\min_{E_c(\cdot),D(\cdot,\cdot)} L = L_{\text{recon}} + \lambda L_{\text{content}}, \quad (5)$$

where

$$\begin{aligned} L_{\text{recon}} &= \mathbb{E}[\|\hat{X}_{1\to1} - X_1\|_2^2], \\ L_{\text{content}} &= \mathbb{E}[\|E_c(\hat{X}_{1\to1}) - C_1\|_1]. \end{aligned} \quad (6)$$

As it turns out, this simple training scheme is sufficient to produce the ideal distribution-matching voice conversion, as will be shown in the next section.

3.3. Why does it work?

We will formally show this autoencoder-based training scheme is able to achieve ideal voice conversion (Eq. (2)). The secret recipe is to have a proper information bottleneck. We will first state the theoretical guarantee and then present an intuitive explanation.

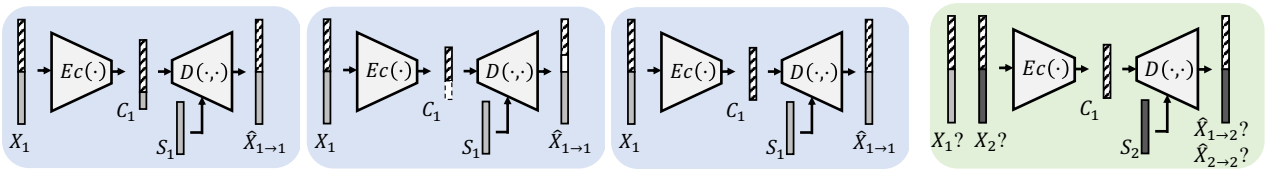
The following theorem characterizes the theoretical guarantee of our proposed framework.

Theorem 1. *Consider the autoencoder framework depicted in Eqs. (3) and (4). Given the following assumption:*

1. The speaker embedding of different utterances of the same speaker is the same. Formally, if $U_1 = U_2$, $E_s(X_1) = E_s(X_2)$.

2. The speaker embedding of different speakers is different. Formally, if $U_1 \neq U_2$, $E_s(X_1) \neq E_s(X_2)$.

[1] X_1' and X_1 can be the same or different.



(a) Bottleneck too wide (b) Bottleneck too narrow (c) Bottleneck just right (d) Conversion

Figure 2. An intuitive explanation of how AUTOVC works. The target speaker is the same as the source speaker during training ((a)-(c)), and different during the actual conversion ((d)). Each speech segments contains two types of information: the speaker information (solid) and content information (striped). (a) When the bottleneck is too wide, the content embedding will contain some source speaker information. (b) When the bottleneck is too narrow, the content information is lost, which leads to imperfect reconstruction. (c) When the bottleneck is just right, perfect reconstruction is achievable, *and* the content embedding contains no source speaker information. (d) During the actual conversion, the output should contain no information about the source speaker, so the conversion quality should be as high as if it were doing self-reconstruction.

3. $\{X_1(1:T)\}$ *is an ergodic stationary order-*τ *Markov process with bounded second moment,* i.e.

$$p_{X_1(t)}(\cdot|X_1(1:t-1),U_1) = p_{X_1(t)}(\cdot|X_1(t-\tau:t-1),U_1). \tag{7}$$

Further assume X_1 *has finite cardinality.*

4. Denote n *as the dimension of* C_1*. Then* $n = \lfloor n^* + T^{2/3} \rfloor$*, where* n^* *is the optimal coding length of* $p_{X_1}(\cdot|U_1)$[2]*.*

Then the following holds. For each T*, there exists a content encoder* $E_c^*(\cdot;T)$ *and a decoder* $D^*(\cdot,\cdot;T)$*,* s.t. $\lim_{T\to\infty} L = 0$*, and*

$$\lim_{T\to\infty} \frac{1}{T} KL\left(p_{\hat{X}_{1\to 2}}(\cdot|u_2,z_1)||p_X(\cdot|U=u_2,Z=z_1)\right) = 0, \tag{8}$$

where $KL(\cdot||\cdot)$ *denotes the KL-divergence.*

The conclusion of Thm. 1 can be interpreted as follows. If the number of frames T is large enough, and if the bottleneck dimension n is properly set, then the global optimizer of the loss function in Eq. (5) would approximately satisfy the ideal conversion property in Eq. (2). This conclusion is quite strong, because a major justification of applying GAN to style transfer, despite all its hassles, is that it can ideally match the distribution of the true samples from the target domain. Now Thm. 1 conveys the following message: to achieve the desired distribution matching, an autoencoder is all you need. The formal proof of Thm. 1 will be presented in the appendix. Here, we will present an intuitive explanation, which is also the gist of our proof. The basic idea is that the bottleneck dimension of the content encoder needs to be set such that it is just enough to code the speaker independent information.

As shown in Fig. 2, speech contains two types of information: the speaker information (shown as solid color) and the speaker-independent information (shown as striped), which we will refer to as the content information[3]. Suppose the bottleneck is very wide, as wide as the input speech X_1. The most convenient way to do self-reconstruction is to copy X_1 as is to the content embedding C_1, and this will guarantee a perfect reconstruction. However as the dimension of C_1 decreases, C_1 is forced to lose some information. Since the autoencoder attempts to achieve perfect reconstruction, it will choose to lose speaker information because the speaker information is already supplied in S_1. In this case, perfect reconstruction is still possible, but the C_1 may contain some speaker information, as shown in Fig. 2(a).

On the other hand, if the bottleneck is very narrow, then the content encoder will be forced to lose so much information that not only the speaker information but also the content information is lost. In this case, the perfect reconstruction is impossible, as shown in Fig. 2(b).

Therefore, as shown in Fig. 2(c), when the dimension of C_1 is chosen such that the dimension reduction is just enough to get rid of all the speaker information but no content information is harmed, we have reached our desirable condition, under which two important properties hold:

1. Perfect reconstruction is achieved.

2. The content embedding C_1 does not contain any information of the source speaker U_1, which we refer to as *speaker disentanglement*.

We will now show by contradiction how these two properties imply an ideal conversion. Suppose when AUTOVC is performing an actual conversion (source and target speakers are different), the quality is low, or does not sound like the target speaker at all. By property 1, we know that the reconstruction (source and target speakers are the same) quality is high. However, according to Eq. (3), the output speech $\hat{X}_{1\to 2}$ can only access C_1 and S_2, both of which do not contain any information of the source speaker U_1.

[2]From the assumpion in Eq. (1), n^* is assumed to be a constant regardless of U_1

[3]The speaker-independent information includes but is not limited to the content information in Z, but for convenience, we will refer to the speaker-independent information as content information.

In other words, from the conversion output, one can never tell if it is produced by self-reconstruction or conversion, as shown in Fig. 2(d). If the conversion quality is low, but the reconstruction quality is high, one will be able to distinguish between conversion and reconstruction above chance, which leads to a contradiction.

4. AUTOVC Architecture

As shown in Fig. 3, AUTOVC consists of three major modules: a speaker encoder, a content encoder, a decoder. AUTOVC works on the speech mel-spectrogram of size N-by-T, where N is the number of mel-frequency bins and T is the number of time steps (frames). A spectrogram inverter is introduced to convert the output mel-spectrogram back to the waveform, which will also be detailed in this section.

4.1. The Speaker Encoder

According to assumptions 1 and 2 in Thm. 1, the goal of the speaker encoder is to produce the same embedding for different utterances of the same speaker, and different embeddings for different speakers. For conventional many-to-many voice conversion, the one-hot encoding of speaker identities suffices. However, in order to perform zero-shot conversion, we need to apply an embedding that is generalizable to unseen speakers. Therefore, inspired by (Jia et al., 2018), we follow the design in (Wan et al., 2018). As shown in Fig. (3)(b), the speaker encoder consists of a stack of two LSTM layers with cell size 768. Only the output of the last time is selected and projected down to dimension 256 with a fully connected layer. The resulting speaker embedding is a 256-by-1 vector. The speaker encoder is pre-trained on the GE2E loss (Wan et al., 2018) (the softmax loss version), which maximizes the embedding similarity among different utterances of the same speaker, and minimizes the similarity among different speakers. Therefore, it is very consistent with assumptions 1 and 2 in Thm. 1.

In our implementation, the speaker encoder is pre-trained on the combination of VoxCeleb1 (Nagrani et al., 2017) and Librispeech (Panayotov et al., 2015) corpora, where there are a total of 3549 speakers.

4.2. The Content Encoder

As shown in Fig. 3(a), the input to the content encoder is the 80-dimensional mel-spectrogram of X_1 concatenated with the speaker embedding, $E_s(X_1)$, at each time step. The concatenated features are fed into three 5×1 convolutional layers, each followed by batch normalization and ReLU activation. The number of channels is 512. The output then passes to a stack of two bidirectional LSTM layers. Both the forward and backward cell dimensions are 32, so their combined dimension is 64.

As a key step of constructing the information bottleneck, both the forward and backward outputs of the bidirectional LSTM are downsampled by 32. The downsampling is performed differently for the forward and backward paths. For the forward output, the time steps $\{0, 32, 64, \cdots\}$ are kept; for the backward output, the time steps $\{31, 63, 95, \cdots\}$ are kept. Figs. 3(e) and (f) also demonstrate how the downsampling is performed (for the ease of demonstration, the downsampling factor is set to 3). The resulting content embedding is a set of two 32-by-$T/32$ matrices, which we will denote $C_{1\rightarrow}$ and $C_{1\leftarrow}$ respectively. The downsampling can be regarded as dimension reduction along the temporal axis, which, together with the dimension reduction along the channel axis, constructs the information bottleneck.

4.3. The Decoder

The architecture of the decoder is inspired by (Shen et al., 2018); and is shown in Fig. 3(c). First, the content and speaker embeddings are both upsampled by copying to restore to the original temporal resolution. Formally, denotes the upsampled features as $U_{\rightarrow}$ and $U_{\leftarrow}$ respectively. Then

$$\begin{aligned} U_{\rightarrow}(:, t) &= C_{1\rightarrow}(:, \lfloor t/32 \rfloor) \\ U_{\leftarrow}(:, t) &= C_{1\leftarrow}(:, \lfloor t/32 \rfloor), \end{aligned} \tag{9}$$

where $(:, t)$ denotes indexing the t-th column. Figs. 3(e) and (f) also demonstrate the copying. The underlying intuition is that each embedding at each time step should contain both past and future information. For the speaker embedding, simply copy the vector T times.

Then, the upsampled embeddings are concatenated and fed into three 5×1 convolutional layers with 512 channels, each followed by batch normalization and ReLU, and then three LSTM layers with cell dimension 1024. The outputs of the LSTM layer are projected to dimension 80 with a 1×1 convolutional layer. This projection output is the initial estimate of the converted speech, denoted as $\tilde{X}_{1\rightarrow 2}$.

In order to construct the fine details of the spectrogram better on top of the initial estimate, we introduce a post-network after the initial estimate, as introduced in Shen et al. (2018). The post network consists of five 5×1 convolutional layers, where batch normalization and hyperbolic tangent are applied to the first four layers. The channel dimension for the first four layers is 512, and goes down to 80 in the final layer. We will refer to the output of the post-network as the residual signal, denoted as $R_{1\rightarrow 2}$. The final conversion result is produced by adding the residual to the initial estimate, *i.e.*

$$\hat{X}_{1\rightarrow 2} = \tilde{X}_{1\rightarrow 2} + R_{1\rightarrow 2}. \tag{10}$$

During training, reconstruction loss is applied to both the initial and final reconstruction results. Formally, in addition to

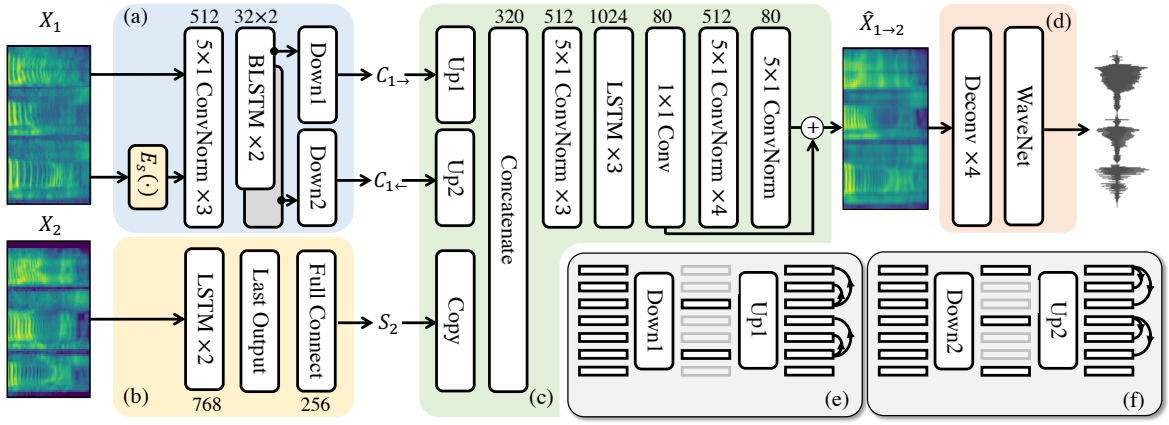


Figure 3. AUTOVC architecture. The number above each block represents the cell/output dimension of the structure. ConvNorm denotes convolution followed by batch normalization. BLSTM denotes bi-directional LSTM, whose white block denotes forward direction, and grey block denotes backward direction. (a) The content encoder. The $E_s(\cdot)$ module is of the same architecture as in (b). (b) The style encoder. (c) The decoder. (d) The spectrogram inverter. (e) and (f) demonstrate the downsampling and upsampling of the forward and backward outputs of the Bi-directional LSTM, using a up/downsampling factor of 3 as an example. The real up/downsampling factor is 32. The lightened feature denotes that they are removed; the arrows denote copying the feature at the arrow origin to the destination.

the loss specified in Eq. (5), we add an initial reconstruction loss defined as

$$L_{\text{recon0}} = \mathbb{E}[\|\tilde{X}_{1\rightarrow 1} - X_1\|_2^2], \tag{11}$$

where $\tilde{X}_{1\rightarrow 1}$ is the reciprocal of $\tilde{X}_{1\rightarrow 2}$ in the reconstruction case, *i.e.* when $U_2 = U_1$. The total loss becomes

$$\min_{E_c(\cdot), D(\cdot,\cdot)} L = L_{\text{recon}} + \mu L_{\text{recon0}} + \lambda L_{\text{content}}. \tag{12}$$

Although Eq. (12) deviates from Eq. (5), on which Thm. 1 rests, we found empirically that this improves convergence and does not harm the performance.

4.4. The Spectrogram Inverter

We apply the WaveNet vocoder as introduced in Van Den Oord et al. (2016), which consists of four deconvolution layers. In our implementation, the frame rate of the mel-spetrogram is 62.5 Hz and the sampling rate of speech waveform is 16 kHz. So the deconvolution layers will upsample the spectrogram to match the sampling rate of the speech waveform. Then, a standard 40-layer WaveNet conditioning upon the upsampled spectrogram is applied to generate the speech waveform. We pre-trained the WaveNet vocoder using the method described in Shen et al. (2018) on the VCTK corpus.

5. Experiments

In this section, we will evaluate AUTOVC on many-to-many voice conversion tasks, and empirically validate the assumptions of the AUTOVC framework. We strongly encourage readers to listen to the demos[4].

[4]https://auspicious3000.github.io/autovc-demo/

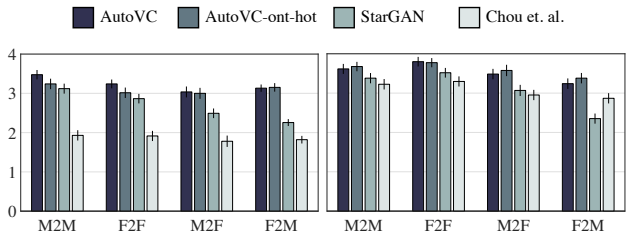
5.1. Configurations

The evaluation is performed on the VCTK corpus (Veaux et al., 2016), which contains 44 hours of utterances from 109 speakers. Each speaker reads a different set of sentences, except for the rainbow passage[5] and the elicitation paragraph. So the conversion setting is non-parallel. Depending on the conversion tasks, different subsets of speakers were selected. The data of each speaker is then partitioned into training and test sets by 9:1. AUTOVC is trained with a batch size of two for 100k steps, using the ADAM optimizer. The speaker embedding is generated by feeding 10 two-second utterances of the same speaker to the speaker encoder and averaging the resulting embeddings. The weights in Eq. (12) are set to $\lambda = 1$, $\mu = 1$.

We performed two subjective tests on Amazon Mechanical Turk (MTurk)[6]. In the first test, called the mean opinion score (MOS) test, the subjects are presented with converted utterances. For each utterance, the subjects are asked to assign a score of 1-5 on the naturalness on the converted speech. In the second test, called the similarity test, the subjects are presented with pairs of utterances. In each pair, there is one converted utterance, and one utterance from the target speaker uttering the same sentence. For each pair, the subjects are asked to assign a score of 1-5 on the voice similarity. We follow the design in Wester et al. (2016) to cue the subjects to judge if the speakers are the same, and how confident they are with their judgment. Thus the similarity score of 5 corresponds to the same speaker with high confidence, and 1 corresponds to different speakers

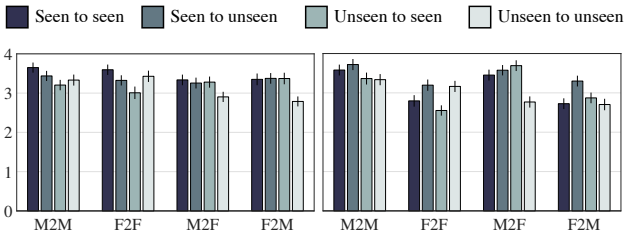
[5]http://web.ku.edu/ idea/readings/rainbow.htm

[6]https://www.mturk.com/



(a) MOS (b) Similarity

Figure 4. Subjective evaluation results for traditional conversion. The error bars denote 95% confidence interval.



(a) MOS (b) Similarity

Figure 5. Subjective evaluation results for zero-shot conversion. The error bars denote 95% confidence interval.

with high confidence. The subjects are explicitly asked to focus on the voice rather than intonation and accent.

5.2. Traditional Many-to-Many Conversion

Traditional many-to-many conversion task performs conversion only on speakers seen in the training corpus. Two baselines are compared with AUTOVC, which we name StarGAN-VC (Kameoka et al., 2018b) and Chou et al. (2018). Both baselines are current state-of-the-arts in non-parallel many-to-many voice conversion. For Chou et al. (2018), we use the original implementation[7] and its pre-trained model, which is trained on 20 speakers in the VCTK corpus. For fair comparison, the other models are trained on the same 20 speakers. Note that the training/test sets are partitioned differently from the Chou et al. pre-trained model, so we are giving the Chou et al. baseline an unfair advantage of seeing part of the test utterance during training. We use the open-source implementation for StarGAN-VC[8].

AUTOVC uses the speaker embeddings produced by the speaker encoder, while the baselines only use the one-hot embeddings of the speakers. To avoid unfair comparison and study if the performance advantage of AUTOVC simply comes from the speaker embeddings, we implement another version of AUTOVC, called AUTOVC-ONE-HOT, that also uses one-hot embeddings of the speakers.

To construct the utterances for the MTurk evaluation, 10 speakers, 5 male and 5 female, are randomly chosen from the 20 speakers in the training set. We then produce $10 \times 9 = 90$ conversions by converting a test utterance of each of the 10 speakers to each of the 10 speakers' voice. Each test unit, called HIT, contains conversion results of the same source-target speaker pair of the three algorithms, so there are 100 HITs in total. Each HIT is assigned to 10 subjects.

Fig. 4(a) presents the MOS scores, and Fig. 4(b) presents the similarity scores. We are dividing the audio into four gender groups, male to male, male to female, female to male and female to female, and summarize the scores within each gender group. As shown in Fig. fig:exper1(a), the perceptual quality of the speech generated by AUTOVC is much better than the baselines'. The MOS scores of AUTOVC are above 3 for all groups, whereas those for the baselines almost all fall below 3. To give readers a better idea of what this means. Notice that the MOS for 16kHz natural speech is around 4.5. The MOS scores of the current state-of-the-art speech synthesizers are between 4 and 4.5 (Shen et al., 2018; Arik et al., 2017). The highest score in 2016 Voice Conversion Challenge (Wester et al., 2016) for *parallel* conversion is 3.8 for same-gender conversions, and 3.2 for cross-gender conversion. Therefore, our subjective evaluation results show that AUTOVC approaches the performance of parallel conversion systems in terms of naturalness, and is much better than existing non-parallel conversion systems.

In terms of similarity, AUTOVC also out-performs the baselines. Note that for the baseline algorithms, there is a significant degrade from same-gender conversion to cross-gender conversion, but AUTOVC algorithms do not display such a degrade. Finally, there is no significant difference between AUTOVC and AUTOVC-ONE-HOT, which implies that the performance gain of AUTOVC does not result from the use of the speaker encoder.

5.3. Zero-Shot Conversion

Now we are ready to go beyond the traditional conversion task towards zero-shot conversion, where the target speakers are absent in the training set and only a few (20 seconds) utterances of each target speaker are available for reference. Since there are no zero-shot conversion baselines, we will compare the results within AUTOVC.

The experiment settings are almost the same as in section 5.2, except that the training set is expanded to 40 speakers to improve the generalizability to unseen speakers. 10 seen speakers 10 unseen speakers are selected for MTurk evaluation, so there are a total of 400 source-target speaker pairs, each producing one conversion utterance. Each HIT contains four utterances, summing up to 100 HITs in total.

[7] https://github.com/jjery2243542/voice_conversion

[8] https://github.com/liusongxiang/StarGAN-Voice-Conversion

Table 1. Assessment of the reconstruction quality and speaker disentanglement of AUTOVC.

	Narrow	AUTOVC	Wide
Recon. Error	34.6	8.59	3.85
Class. Acc.	7.50%	12.0%	70.5%

Each HIT is assigned to 10 subjects.

Fig. 5 presents the scores. There are three observations. First, for conversions among seen speakers, the performance is comparable to that in section 5.2. Note that in this experiment, AUTOVC is trained on 40 speakers, which doubles the number of speakers used in the experiment in section 5.2. Therefore, this comparable performance on seen speakers indicates that AUTOVC is scalable to a large number of speakers in the training set.

Second, in terms of MOS score, AUTOVC shows good generalizations to unseen speakers, with the MOS score exceeding 3 in most settings. This means, even for unseen speakers, AUTOVC is still able to outperform most existing non-parallel conversion algorithms.

Finally, in terms of the similarity score, there is an interesting observation that as long as seen speakers are included in either side of the conversions, the performance is comparable. There is a significant gap between conversions from unseen speakers to unseen speakers and the rest of the paradigms. Nevertheless, even for conversions within unseen speakers, which is the most challenging case, the similarity scores are still very competitive, which demonstrates AUTOVC's competence in zero-shot conversion.

5.4. Bottleneck Dimension Analysis

Our theoretical justifications for the proposed style transfer autoencoder lies in the claim that the bottleneck dimension affects perfect reconstruction and disentanglement of content code and source speaker information, and that there exists a desirable bottleneck dimension where both properties hold (Fig. 2). In this section, we will empirically validate this claim.

We measure AUTOVC's reconstruction quality and the degree of disentanglement between the content code and the source speaker information. The reconstruction quality is measured by the ℓ_2-norm of reconstruction error in the training set. Lower reconstruction error means higher reconstruction quality. The disentanglement is measured by training a speaker classifier on the content code and computing the classification accuracy on the training set. Higher classification accuracy means poorer disentanglement. The speaker classifier consists of 3 fully-connected layers with 2,048, 1,024 and 1,024 hidden nodes respectively in each layer and softplus activation. The output activation is softmax and the training loss is cross entropy. The model architecture and experiment setting follow those in section 5.3, so the speaker classification is on the 40 seen speakers.

As references, we introduce two anchor models. The first model, which we name the "too narrow" model, reduces the dimensions of $C_{1\rightarrow}$ and $C_{1\leftarrow}$ from 32 to 16, and increases the downsampling factor from 32 to 128 (note that higher downsampling factor means lower temporal dimension). The second model, which we name the "too wide" model, increases the dimensions of $C_{1\rightarrow}$ and $C_{1\leftarrow}$ to 256, and decreases the sampling factor to 8, and λ is set to 0. Supposedly, according to Fig. 2, the "too narrow" model should have low classification accuracy (good disentanglement) but high reconstruction error (poor reconstruction). The "too wide" model should have low reconstruction error (good reconstruction) but high classification accuracy (poor disentanglement). The normal AUTOVC model should have both low reconstruction error (good reconstruction) and low classification accuracy (good disentanglement).

Table. 1 shows the reconstruction error and speaker classification accuracy for the three models. As expected, as the bottleneck dimension decreases, the reconstruction error increases and the classification accuracy decreases. What is interesting is that the normal AUTOVC model does strike a good balance, with reconstruction error almost as low as the "too wide" model and the classification accuracy almost as low as the "too narrow" model. It is worth mentioning that Chou et al. (2018) explicitly perform adversarial training to enforce speaker disentanglement. A similar classification experiment to test disentanglement is performed, and the classification accuracy is 45.1% on 20 speakers after the adversarial training is applied. In order to fairly compare with this result, we also perform a speaker classification test on the same 20 speakers, and the classification accuracy is 14.2%. This result shows that bottleneck dimension tuning on speaker disentanglement is more effective than the more sophisticated adversarial training.

6. Conclusion

In this paper, we have proposed AUTOVC, a non-parallel voice conversion algorithm that significantly outperforms the existing state-of-the-art, and that is the first to perform zero-shot conversions. In sharp contrast to its performance advantage is its simple autoencoder structure that trains only on self-reconstruction, and a bottleneck tuning to balance between reconstruction quality and speaker disentanglement. In an era of building increasingly sophisticated algorithms for style transfer, our theoretical justification and the success of AUTOVC suggest that it is time to return to simplicity, because sometimes an autoencoder with a careful bottleneck design is all you need to make a difference.

References

Arik, S. O., Chrzanowski, M., Coates, A., Diamos, G., Gibiansky, A., Kang, Y., Li, X., Miller, J., Ng, A., Raiman, J., et al. Deep voice: Real-time neural text-to-speech. *arXiv preprint arXiv:1702.07825*, 2017.

Atalla, C., Tam, B., Song, A., and Cottrell, G. Look ma, no GANs! image transformation with modifAE, 2019. URL https://openreview.net/forum?id=B1ethsR9Ym.

Biadsy, F., Weiss, R. J., Moreno, P. J., Kanvesky, D., and Jia, Y. Parrotron: An end-to-end speech-to-speech conversion model and its applications to hearing-impaired speech and speech separation. *arXiv preprint arXiv:1904.04169*, 2019.

Chou, J.-c., Yeh, C.-c., Lee, H.-y., and Lee, L.-s. Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations. *arXiv preprint arXiv:1804.02812*, 2018.

Donahue, C., McAuley, J., and Puckette, M. Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*, 2018.

Fan, Z.-C., Lai, Y.-L., and Jang, J.-S. R. SVSGAN: Singing voice separation via generative adversarial network. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 726–730. IEEE, 2018.

Fang, F., Yamagishi, J., Echizen, I., and Lorenzo-Trueba, J. High-quality nonparallel voice conversion based on cycle-consistent adversarial network. *arXiv preprint arXiv:1804.00425*, 2018.

Gao, Y., Singh, R., and Raj, B. Voice impersonation using generative adversarial networks. *arXiv preprint arXiv:1802.06840*, 2018.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Heigold, G., Moreno, I., Bengio, S., and Shazeer, N. End-to-end text-dependent speaker verification. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 5115–5119. IEEE, 2016.

Hosseini-Asl, E., Zhou, Y., Xiong, C., and Socher, R. A multi-discriminator cyclegan for unsupervised non-parallel speech domain adaptation. *arXiv preprint arXiv:1804.00522*, 2018.

Hsu, C.-C., Hwang, H.-T., Wu, Y.-C., Tsao, Y., and Wang, H.-M. Voice conversion from non-parallel corpora using variational auto-encoder. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2016 Asia-Pacific*, pp. 1–6. IEEE, 2016.

Hsu, C.-C., Hwang, H.-T., Wu, Y.-C., Tsao, Y., and Wang, H.-M. Voice conversion from unaligned corpora using variational autoencoding Wasserstein generative adversarial networks. *arXiv preprint arXiv:1704.00849*, 2017.

Huang, W.-C., Hwang, H.-T., Peng, Y.-H., Tsao, Y., and Wang, H.-M. Voice conversion based on cross-domain features using variational auto encoders. *arXiv preprint arXiv:1808.09634*, 2018.

Jia, Y., Zhang, Y., Weiss, R. J., Wang, Q., Shen, J., Ren, F., Chen, Z., Nguyen, P., Pang, R., Moreno, I. L., et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *arXiv preprint arXiv:1806.04558*, 2018.

Kameoka, H., Kaneko, T., Tanaka, K., and Hojo, N. Acvae-vc: Non-parallel many-to-many voice conversion with auxiliary classifier variational autoencoder. *arXiv preprint arXiv:1808.05092*, 2018a.

Kameoka, H., Kaneko, T., Tanaka, K., and Hojo, N. Stargan-vc: Non-parallel many-to-many voice conversion with star generative adversarial networks. *arXiv preprint arXiv:1806.02169*, 2018b.

Kaneko, T. and Kameoka, H. Parallel-data-free voice conversion using cycle-consistent adversarial networks. *arXiv preprint arXiv:1711.11293*, 2017.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pp. 3581–3589, 2014.

Nachmani, E. and Wolf, L. Unsupervised singing voice conversion. *arXiv preprint arXiv:1904.06590*, 2019.

Nagrani, A., Chung, J. S., and Zisserman, A. Voxceleb: a large-scale speaker identification dataset. In *INTERSPEECH*, 2017.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, April 2015. doi: 10.1109/ICASSP.2015.7178964.

Pascual, S., Bonafonte, A., and Serra, J. Segan: Speech enhancement generative adversarial network. *arXiv preprint arXiv:1703.09452*, 2017.

Saito, Y., Ijima, Y., Nishida, K., and Takamichi, S. Non-parallel voice conversion using variational autoencoders conditioned by phonetic posteriorgrams and d-vectors. In *Proc. ICASSP*, pp. 5274–5278, 2018.

Shen, J., Pang, R., Weiss, R. J., Schuster, M., Jaitly, N., Yang, Z., Chen, Z., Zhang, Y., Wang, Y., Skerrv-Ryan, R., et al. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783. IEEE, 2018.

Subakan, Y. C. and Smaragdis, P. Generative adversarial source separation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 26–30. IEEE, 2018.

Van Den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *CoRR abs/1609.03499*, 2016.

Veaux, C., Yamagishi, J., MacDonald, K., et al. Superseded-CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit. 2016.

Wan, L., Wang, Q., Papir, A., and Moreno, I. L. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4879–4883. IEEE, 2018.

Wester, M., Wu, Z., and Yamagishi, J. Analysis of the voice conversion challenge 2016 evaluation results. In *INTERSPEECH*, pp. 1637–1641, 2016.

Xie, F.-L., Soong, F. K., and Li, H. A KL divergence and DNN-based approach to voice conversion without parallel training sentences. In *INTERSPEECH*, pp. 287–291, 2016.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*, 2017.