École Polytechnique Fédérale de Lausanne

and

Logitech Europe S.A.

Master Thesis

Speech Style Transfer

by Oriol Barbany Mayor

Supervisors:

Prof. Martin Jaggi
Master project supervisor - Machine Learning and Optimization Laboratory, EPFL

Dr. Milos Cernak
In-company supervisor - CTO Office, Logitech

August 14, 2020

Science is what we understand well enough to explain to a computer.
Art is everything else we do.
— Donald Knuth

Acknowledgments

First of all, I would like to thank all my colleagues at Logitech. In particular, I would like to thank each of the interns; Alexandru, Beryl, Gaia, Gauthier, Giovanna, Gregoire, Joachim, Kelian, Lionel, Lorenzo, Luisa, Martina, both Maximes, Michael, Paul, Rayan, Virgile, Yassine, and Yi-Shiun. This internship has flown by so fast for all of us. Out of the 6 months, we spent less than a couple of them doing regular office life. Despite the short time I spent with all you guys, I really got to know how brilliant and funny you are. I take good memories from this time, and I am delighted to have taken part in this team.

I would like to express my sincere gratitude to my Logitech supervisor, Dr. Milos Cernak, for his trust, expertise, and support with my future career. The weekly meetings that he scheduled with all the interns working in Machine Learning and audio really incentivized collaboration. Each of us knew the others' projects, which helped us take ideas from them and discuss the similarities. Thanks to my EPFL supervisor Prof. Martin Jaggi for his helpful comments in the project follow-up meetings and the predisposition to help me pursue a Ph. D.

I would also like to acknowledge Kaizhi Qian for providing the full code used as a starting point for this project. Even if this code was not open-source, I really appreciate that he was willing to share it for my project.

I am particularly grateful for the opportunities that Logitech and Astro Gaming gave me. During my internship, I got all the resources I needed. I even got the chance to share my knowledge in a webinar for all the company! For this latter, I would like to acknowledge Arash, Pablo, and Thomas from the CTO Office in Logitech for their help with managing resources and the project presentations. Finally, I could not finish the acknowledgments to Logitech without mentioning the enormous support and predisposition of Jean-Michel Chardon and Melanie Poget.

This document closes one of the most challenging and enriching chapters of my life. Coming to EPFL was definitely one of the best decisions I have ever taken. The quality of the professors, facilities, and the courses offered are really top-notch. In these two years, I learned (and, of course, worked) a lot. Given the openness of the study plan of my master's degree, I had the opportunity of exploring almost any of the branches of mathematics and computer science. This helped me realize my research interests and that I want to follow a career in research.

I came to Switzerland alone, and I will leave with a lot of friends. Some of them were there when I had to study, and some of them were there when I wanted to have fun. But I am glad to say that all of them were there to support me in any case. Special thanks to Manuel Cherep for his technical knowledge and the help with the bureaucratic stuff I have encountered during these years. Thanks to David Álvarez from my Bachelor's, who helped me review and improve this document.

Last but not least, I would like to switch to Catalan so my family can avoid using a translator:

Moltes gràcies per les oportunitats que m'heu donat durant tots aquests anys. Especialment al Rafel i la Gemma, els meus pares, que m'han ajudat moralment i econòmicament durant tota aquesta etapa. També gràcies a la Laura, la meva germana, per fer la distància una mica més curta amb l'ajuda de les videotrucades i per fer que quan torno a casa em senti com si sempre visqués amb vosaltres. Moltes gràcies a tots per creure en mi i recolzar-me en totes les meves decisions. Us estimo.

Lausanne, September 6, 2020 Oriol Barbany Mayor

Abstract

The problem of style transfer consists in transferring the style from one signal to another while preserving the latter's content. This project explores the applications of style transfer techniques to speech signals. In particular, such techniques are used to address Voice Conversion (VC). This problem can be formulated with the style transfer framework and consists of changing the speaker identity (style) of a speech signal at will while preserving the same linguistic information (content).

Style transfer is an inherently ill-posed problem; i.e., there is not a unique solution. Moreover, there are no standardized objective measures to evaluate the results. The effect of this is twofold. Firstly, the lack of such metrics hinders the training process. Secondly, it is hard to benchmark different methods.

The first problem is tackled by using an AutoEncoder (AE) architecture. The raw speech is mapped to a lower-dimensional latent space where linguistic and speaker content is separated. During training, the model learns to reconstruct the original raw speech from this representation. When performing VC, the latent representation is modified to match the target speaker. This work presents two variants of this approach named FastVC and PhonetVC.

The problem with comparing to the state-of-the-art is solved with the large-scale crowd-sourced perceptual evaluations performed in the Voice Conversion Challenge. The 2020 edition of this challenge centers in non-parallel VC. In particular, a variant of FastVC was submitted for the cross-lingual task, where the target and source speakers speak different languages. FastVC outperformed the VC Challenge baselines and ranked in the top half of the classification in Mean Opinion Score (MOS) quality results among all the participants.

The unsupervised representations found with FastVC are shown to be speaker-independent and easily mapped to the human phoneme alphabet. The analysis of such representations confirms that encoder-decoder architectures allow disentangling the style and content of a speech signal. Overall, FastVC offers high-quality results for the task of VC while providing speedy conversions.

Contents

List of Figures

List of Tables

Acronyms

AE AutoEncoder

AR Auto-Regressive

ASR Automatic Speech Recognition

CNN Convolutional Neural Network

COLA Constant OverLap-Add

DTW Dynamic Time Warping

GAN Generative Adversarial Network

GE2E Generalized End-to-End

MFCC Mel-Frequency Cepstral Coefficient

MOS Mean Opinion Score

MSE Mean Squared Error

PCA Principal Component Analysis

PESQ Perceptual Evaluation of Speech Quality

PLLR Phonological Log-Likelihood Ratio

PPG Phonetic PosterioGram

RNN Recurrent Neural Network

SGD Stochastic Gradient Descent

STFT Short-Time Fourier Transform

t-SNE t-Distributed Stochastic Neighbor Embedding

TTS Text-to-Speech

VAD Voice Activity Detector

VAE Variational AutoEncoder

VC Voice Conversion

VQ Vector Quantization

1 Introduction

One of the main strengths of machine learning is its data-driven nature. Models can solve a wide variety of tasks and be implemented in very different setups with successful results. Moreover, with transfer learning, it is relatively easy to reuse work in similar domains. One could thus solve a problem to some extent from some scientific domain without knowing its underlying meaning. Nonetheless, extensive knowledge of that domain is needed to improve the solutions in the form of inductive biases and priors.

Overall, one can benefit a lot from the machine learning research community by adapting previously successful solutions to many problems in almost any scientific domain. In particular, the computer vision community is actively developing new algorithms and architectures that have been proven to be beneficial to other domains such as speech. One clear example of such architectures are Convolutional Neural Networks (CNNs), which were first trained with backpropagation by LeCun et al. (1989). This architecture was designed to exploit the similarity of neighboring pixels in natural images and achieve the desirable equivariance property. A common approach for audio applications is to apply such architectures to the spectrogram, which corresponds to the squared magnitude of the Short-Time Fourier Transform (STFT) of the input waveform. The result of the STFT is a 2-dimensional complex signal representing time and frequency. Thus, the spectrogram, by being a real 2-dimensional signal, can be interpreted as an image. The spectrogram has smooth transitions, which make the locality assumption of CNNs valid. However, the equivariance property is no longer desirable, since shifting of the spectrogram is translated into a time delay or a frequency modulation of the signal. Nevertheless, even if the inductive bias of CNNs does not precisely fit the nature of spectrograms, this architecture is still widely used for such signals.

The problem of working with spectrograms becomes harder when it comes to invertibility. One can recover the original signal using the inverse STFT from both the magnitude and phase of the STFT when the Constant OverLap-Add (COLA) property is satisfied (Smith, 2011). However, the phase component of a speech signal is tricky to model because it has a noisy structure and a circular nature provided that it is an angle. Thus, the assumptions of CNNs are far from being satisfied with this signal. One common approach is to use the classical method introduced by Griffin and Jae Lim (1984) to recover the phase from the spectrogram. Even if the phase is essentially random, relative phase differences over time matter perceptually. Due to this latter, de Chaumont Quitry et al. (2019) proposed to learn the time-derivative of the phase and then integrate instead of trying to learn a seemingly random pattern. Results of phase prediction algorithms are still far from giving real phases and usually produces metallic audios. However, Griffin and Jae Lim (1984) is still used in some recent works.

The more recent approaches to the problem of spectrogram inversion have nowadays shifted from recovering the phase and using inverse STFT to relying on conditional generative models that directly produce waveforms. If we model waveforms directly, we are implicitly modeling the phase of its STFT as well, but we do not run into the issues that make the modeling of phase so cumbersome.

Inverting spectrograms that are directly extracted from raw speech is a challenging problem. However, in applications like Text-To-Speech (Wang et al., 2017) and Voice Conversion (VC)

(Qian et al., 2019; Tobing et al., 2019; Pasini, 2019), the spectrogram itself is respectively modeled from text or some speech and speaker features. In such cases, the spectrogram that has to be inverted generally has imperfections (see Section 4.1). The generative model can learn to invert a representation with such imperfections. Moreover, it can also work with lossy spectrogram representations as it is the case for the mel-spectrogram.

The fact that generative models work better than the classical methods for phase recovering and deterministic STFT inversion is mainly due to the inductive biases with which some of the most successful models are endowed. Speech is a signal with dependencies in both neighboring samples and over long time spans on the time domain due to, e.g., the periodicity of voiced speech and the intonation or pitch contour. WaveNet (van den Oord et al., 2016) models such long-term dependencies by introducing stacked causal dilated convolutions, which exponentially augment the receptive field with the number of layers (see Section 2.3.1). The large receptive field allows taking into account more past samples, which allows modeling longer time dependencies without the need to have very deep models. Another approach to model both short and long-term dependencies is to have different modules running at multiple resolutions. This hierarchical structure is implemented in SampleRNN (Mehri et al., 2016) and in the discriminators of Kumar et al. (2019) (see Section 2.3.2). The success of these approaches shows the importance of expert knowledge.

1.1 Style Transfer

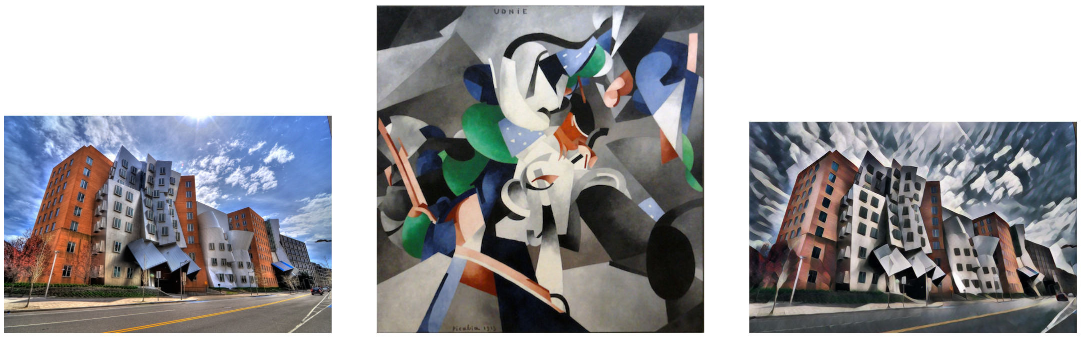
This project tackles the problem of speech style transfer, including the generation of raw speech and hence the previously discussed issues. On top of that, the style transfer problem requires the modification of high-level signal features, making it an even more challenging task. Deep learning is especially suited for this task due to the quality of the representations learned from data. These representations capture latent variables of the signal that can be useful to represent the abstract concepts of content and style of a signal required in the style transfer problem.

Neural style transfer was introduced in Gatys et al. (2015) and has been mostly applied to images (Dumoulin et al., 2016) and videos (Ruder et al., 2016). This task is defined for two input signals, which we will refer to as source and target. The task is then to transfer the target signal's style to the source signal while preserving its content. Therefore, the success in this task requires semantic reasoning about the input signal (Johnson et al., 2016).

The style transfer problem statement includes the abstract terms of content and style, which makes this task loosely defined mathematically speaking. Figure 1.1 depicts an example of image style transfer. The concepts of content and style may be clearer with the preceding example, but they still lack a proper definition.

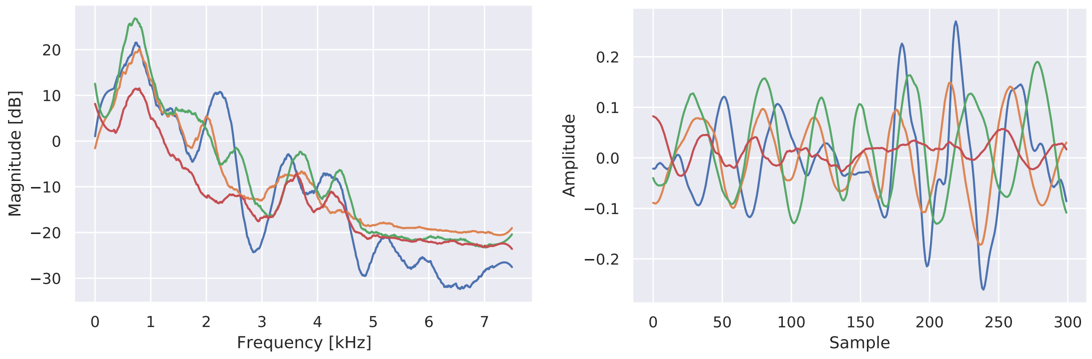
VC can be considered as a case of speech style transfer, where the content refers to the linguistic information of a speech signal, and the style refers to the speaker. The task of VC then consists of modifying a speech signal uttered by some source speaker as another target speaker uttered it. In such a task, the linguistic information is preserved, but speaker-dependent features are changed.

Both style transfer and VC problems are inherently ill-posed; there is not a single correct output. For the case of VC, even if it is easy for a human to identify the concepts of linguistic



(a) Source image. (b) Target image. (c) Stylised image.

Figure 1.1: Example of image style transfer from Engstrom (2016). The stylised image (c) has the content of the source image (a), which is a photograph of the MIT Stata Center, and the style of the target image (b), the artwork Udnie by Francis Picabia. Note that the source and target image don't necessarily have to have the same size.



(a) Smoothed spectrum. (b) First samples in the waveform domain.

Figure 1.2: Comparison between 4 speech signals sampled at 44.1 kHz of 0.1s each with the same speaker uttering the sound `/a/`. All the recordings sound almost identical but are very different in the waveform domain. The frequency representations of such signals are also different but have some similarities such as the formants (local maximums). This motivates the use of frequency representations of the speech signals such as the STFT.

content and speaker style, it is hard to quantify how similar the result is to the target speaker or how unaltered is the linguistic content.

The same content can be uttered by the same speaker in many different ways, as shown in Figure 1.2. Additionally, there are many transformations on the waveform level that do not affect the content at all. Some examples are different timing, intonation, volume, or flipping of the sign of the signal.

The difficulty in assessing VC systems brings other problems. On the one hand, the lack of objective measures hinders choosing a training strategy and an objective function. On the other hand, most of the works in VC only report subjective scores. The subjective score gives an idea of how a system works, but such scores may be biased. Therefore, many papers claim state-of-the-art results, but it is impossible to compare two models if they only report subjective scores.

1.2 Document Structure

Chapter 2 reviews some of the existing solutions for the problems of style transfer, VC, and speech synthesis. The most influential works for this project are more thoroughly analyzed, with particular emphasis on AutoVC and MelGAN (Section 2.1.1 and Section 2.3.2, respectively). This section also provides an overview of the VC challenge in Section 2.4, which serves as a benchmark for this project.

Chapter 3 presents the proposed system for VC, FastVC. This system is an adaptation of the state-of-the-art model AutoVC introduced by Qian et al. (2019). This system is modified to meet the requirements of the VC challenge 2020, where more than 90 research teams competed to achieve the best VC system.

Besides the speech quality and the target similarity, which are the factors evaluated in the VC challenge, this project also centers on achieving faster conversions. This latter is not evaluated in the challenge, and in fact, most of the VC papers do not report the speed of conversion. Nonetheless, achieving fast inference in audio can be even more challenging than generating good conversions due to the high temporal resolution of raw audio.

The speed requirements are met using MelGAN (Kumar et al., 2019), which generates raw speech in a non-Auto-Regressive (AR) fashion. The resulting system presented in Chapter 3 integrates an AutoEncoder (AE) adapted from AutoVC (Qian et al., 2019) and the raw speech generator from Kumar et al. (2019). To achieve real end-to-end training, a simple network that can be initialized to yield exact spectrograms is proposed. The aim is that pre-trained models using (mel-)spectrograms can be easily adapted for end-to-end training. Even if the mel-spectrogram is a ubiquitous representation that considers human perception, more efficient representations that are better suited for a given task can be learned (see Section 3.1).

Chapter 4 compares the proposed models and presents an analysis of the speech representations found by FastVC. Such representations aim at disentangling the linguistic and speaker information and are shown to be speaker-independent and similar to phonemes. The subjective evaluations obtained with the samples submitted to the VC challenge are presented in Section 4.5.

Finally, Chapter 5 wraps up the project and proposes some improvements to our work and future directions for the VC problem.

2 Literature Review

Style transfer requires the extraction of the content and the style of a signal. In Gatys et al. (2015), the hypothesis is that content and style are related to respectively low and high-level representations of the image, as depicted in Figure 2.1. Such representations are respectively found in the feature maps obtained by the first and last layers of a CNN. The model obtaining those feature maps is not explicitly trained for the style transfer problem, but instead, a VGG-Network (Simonyan and Zisserman, 2015) trained on object recognition is used.

Once the style and content are defined, the approach is to optimize over the pixel values of an input image initialized to white noise. The aim is to match both the content and the style of the generated image to that of the source and target image, respectively.

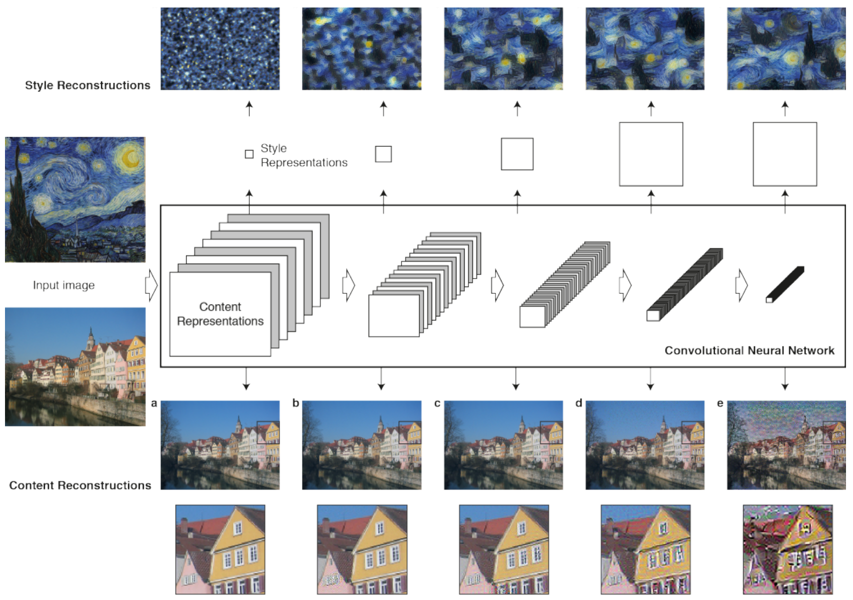
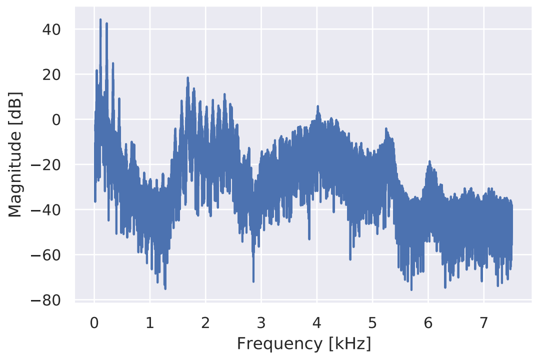


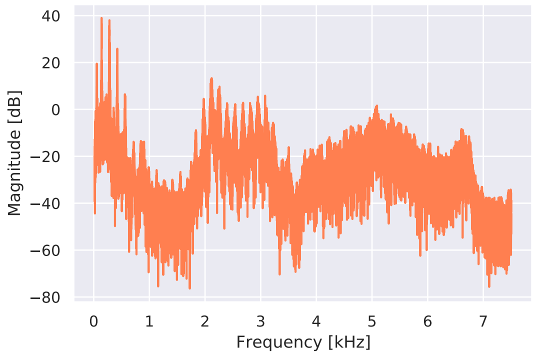
Figure 2.1: The content reconstructions are obtained recovering the input image from the feature maps. The style representations compute correlations between subsets of feature maps. Note that indeed the style representation that better fits the style of the image is obtained using high-level representations, while low-level representations are better at capturing the content.

The main drawback of the approach by Gatys et al. (2015) is that since optimization is performed over the input, the resulting image has to be iteratively computed by performing gradient updates on the combination of content and style losses. Since these losses are different for different styles and contents, inference requires solving an optimization problem for each pair of source and target signals. Several methods differing from optimization-based style transfer have been proposed to compute a fixed mapping function (Johnson et al., 2016; Zhu et al., 2017; Mor et al., 2019; Beckmann et al., 2019). Such mapping functions are especially suited for applications where time or resources are constrained.

VC requires the factorization of speech into linguistic and non-linguistic information to modify this latter at will without changing the linguistic content of the source audio. There are two main frameworks for the data-driven approach to VC. These are parallel and non-parallel VC.



(a) Spectrum of original signal.



(b) Spectrum of original signal shifted 4 semitones up.

Figure 2.2: Pitch shifting with a signal of a speaker uttering a long vowel `/i/`. Note that the spectral envelope is preserved, but the fundamental frequency changes. The pitch shifting algorithm used in this example is the implementation by McFee et al. (2020).

A parallel speech dataset consists of pairs of time-aligned utterances with the same linguistic content but uttered by different speakers. The approach to VC is then to learn the mapping between them (Abe et al., 1988). This kind of data is scarce and the pairs of utterances are typically not perfectly aligned. The usual approach is to use Dynamic Time Warping (DTW) so that each time-aligned frame pair shares the same linguistic information. However, the research community is moving towards non-parallel training.

Non-parallel VC is a more challenging framework, but contrary to parallel VC, it can use virtually any speech dataset for training. The results with this approach are typically worse than parallel VC (Lorenzo-Trueba et al., 2018), which drives some methods first to construct pseudo-parallel datasets using acoustic clustering of source and target speakers (Sundermann et al., 2006).

Taigman et al. (2017) introduced zero-shot voice generation, i.e., the generation of speech from speakers unseen during the training phase. Zero-shot conversion introduces a new problem to VC and can only be performed in multi-speaker models that use data-driven speaker representations. However, this problem is not treated in this work because pre-trained networks can be used to compute such speaker representations as in Qian et al. (2019).

The d-vectors introduced by Jia et al. (2018) are one example of such representations that can be computed for arbitrary long audios and offers reliable representations with only a few seconds of speech. Such representations can be interpreted as a decomposition of each speaker's identity into a vector space whose basis represent different speaker styles or *eigen-voices* (Barbany, 2018, Section 3.3.).

2.1 Conversion function

Although simple conversion functions are capable of changing the speaker's identity, it is insufficient to convert a specific source speaker's voice into another specific target speaker's voice. One of such simple functions is frequency warping, which can change the pitch without altering the envelope as in Figure 2.2.

However, more sophisticated conversion functions are needed to adequately model a nonlinear mapping between source and target voices that need to be developed to convert speaker identity (Toda et al., 2016). Various conversions functions have been proposed using, amongst others, Gaussian Mixture Models (Stylianou et al., 1998), Restricted Boltzmann Machines (Nakashika et al., 2014), or Gaussian process regression with kernel functions (Pilkington et al., 2011). The dominant approach is to model the conversion function with generative models built using Deep Neural Networks, usually called Deep Generative Models.

Generative models learn the underlying distribution from which the input is assumed to be independently drawn from. Speech signals are structured data with a time dimension, and thus the assumption of independence is far from being satisfied. Instead of learning the distribution of samples, generative models learn the conditional distribution of a sample given its previous samples (van den Oord et al., 2016; Mehri et al., 2016). Generating samples only conditioned on the previous samples produce babbling sounds that resemble speech but lack coherent linguistic information. The approach to generate coherent speech is to further condition these models on lexical information so that the generated samples represent the sound that we want to produce.

A conditioning signal can be of various shapes and rates, and we can differentiate between local and global conditioning, which are also referred to as dense and sparse conditioning, respectively (Dieleman, 2020). Regarding speech, the generative models are typically conditioned on phonemes, which are a local conditioning signal running at a lower rate than the raw waveform samples. In multi-speaker settings, the generative models are also conditioned on the speaker's identity, a global conditioning signal. Some popular Deep Generative Models include AR models, the decoder of Variational AutoEncoders (VAEs) and Generative Adversarial Networks (GANs).

AR models are powerful because they can capture correlations, especially local dependencies, between the different elements in a sequence. However, each sample has to be drawn sequentially because the samples' distribution at a given time depends on the previous samples, making the generation slow.

A VAE is a two-stage network. The first stage encodes a given input into a latent representation of lower dimensionality with a known distribution, typically a unit Gaussian. The second stage decodes the latent representation to obtain a reconstruction as similar to the input as possible. In the case where the latent representation is not enforced to match a certain distribution, the network is called an AE.

GANs take a very different approach to capture the data distribution. In this setup, two networks are trained simultaneously: a generator attempting to produce examples that fit the underlying distribution of the data given latent vectors, and a discriminator attempting to differentiate between the real inputs and the generator's outputs. In doing so, the discriminator provides a learning signal for the generator, which enables it to better match the data distribution.

The drawback of adversarial models is that, even if they produce more realistic examples than likelihood-based models, they are worse at capturing the full diversity of the data distribution. This behavior is particularly useful in densely-conditioned settings since, in such cases, the variability is mostly constrained. An example of a densely conditioned setting is Text-to-Speech (TTS): given an excerpt of text, the model should generate a realistic utterance corresponding to that excerpt. In this case, there is no need to generate every possible variation, but one good-sounding utterance is enough. In this setting, realism is more important than diversity, which is already captured by the conditioning signal.

The AE structure allows extracting meaningful latent representations of speech in an unsupervised fashion. These representations are ideally able to capture high-level semantic content from speech such as phoneme identities (Chorowski et al., 2019) and thus could avoid manually labeling speech datasets. This content can be leveraged to generate speaker-independent utterances.

Some of the most successful works in VC rely on generative models like AEs or VAEs that compress the input signal and recover it from the compressed representation. The compressed signal is in a lower-dimensional space, which enforces the model to find compact representations only containing the relevant information needed to reconstruct the signal.

One approach to disentangle the linguistic and speaker information with this method is to make one of the former irrelevant for the reconstruction. In Barbany (2018), this is done by feeding uncompressed speaker information to the decoder. Speaker disentanglement, in this case, can be justified with the redundancy principle (Barlow, 1989). Since the decoder is explicitly conditioned on the speaker identity, the encoder does not have to capture speaker-dependent information in the latents. The desired speaker independence is achieved if the dimension of the latent space satisfies the following trade-off. On the one hand, it has to be sufficiently small to factor out the speaker's information. On the other hand, it has to be large enough to allow for perfect reconstruction and thus capture as much of the input data as possible.

2.1.1 AutoVC

AutoVC was proposed by Qian et al. (2019), and it was the first system to perform zero-shot VC. This system achieves state-of-the-art results in many-to-many VC with non-parallel data. Despite the astonishing results of this approach, the architecture of AutoVC is basic and based on a simple conditional AE. Note that AEs learn to compress and decompress an input signal, so VC does not happen during training. When AutoVC is in conversion mode, both the latent representation obtained by the encoder and the target speaker information are used to generate the output speech.

Let $\mathbf{X}_1$ and $\mathbf{X}_2$ be the log-scale mel-spectrograms of the source and target speech signals. AutoVC is only fed these two signals, so any kind of transcription or time-aligned feature is needed. The converted speech is obtained using four different modules, as depicted in Figure 2.3. Note that during training, both the source and the target inputs are the same, i.e., $\mathbf{X}_2 = \mathbf{X}_1$. In this case, the output of the decoder is denoted as $\widehat{\mathbf{X}}_{1\to 1}$.

Zero-shot VC is achieved using the pre-trained speaker encoder. In particular, the speaker encoder E_s (module (b) in Figure 2.3) is trained by the AutoVC authors using the architecture and the Generalized End-to-End (GE2E) loss described in Wan et al. (2017). The GE2E is particularly suitable for the task of speaker representation because it maximizes the similarity of the embeddings belonging to a given speaker and minimizes the similarity of the embeddings for utterances of different speakers. The speaker encoder network is trained with the GE2E loss alone, and its weights are not updated during the training of AutoVC.

The training objective of AutoVC includes the usual self-reconstruction loss of AEs. In particular, the expected squared ℓ_2 norm between the input and the output is used (2.1).

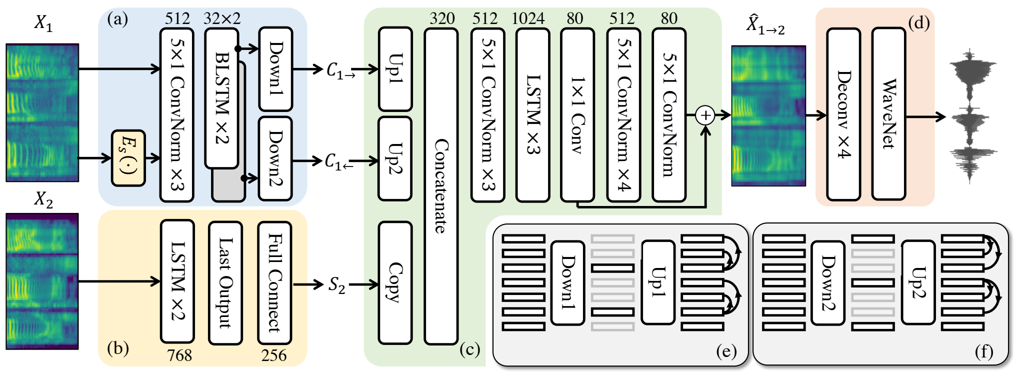


Figure 2.3: AutoVC (Qian et al., 2019) model architecture in conversion mode. The target signal $\mathbf{X}_2$ is fed to the style encoder (b), which outputs the representation of the target speaker, $\mathbf{s}_2 := E_s(\mathbf{X}_2)$. The source signal $\mathbf{X}_1$ is also fed to the style encoder (b) giving $\mathbf{s}_1 := E_s(\mathbf{X}_2)$. The input of the content encoder (a) is the source signal $\mathbf{X}_1$ and the latter source speaker representation $\mathbf{s}_1$. The content encoder applies information bottlenecks to the input and outputs the forward and backward code, denoted as $\mathbf{C}_{1\rightarrow}$ and $\mathbf{C}_{1\leftarrow}$ respectively. The output of the content encoder $(\mathbf{C}_{1\rightarrow}, \mathbf{C}_{1\leftarrow}) := E_c(\mathbf{X}_1, \mathbf{s}_1)$ and the target speaker representation $\mathbf{s}_2$ are fed to the decoder, which outputs the log-scale mel-spectrogram $\widehat{\mathbf{X}}_{1\rightarrow 2} := D(\mathbf{C}_{1\rightarrow}, \mathbf{C}_{1\leftarrow}, \mathbf{s}_2)$, that contains the content of the source signal and the style of the target. This representation is mapped to raw speech with (d). The subfigures (e) and (f) represent the downsampling and upsampling of $\mathbf{C}_{1\rightarrow}$ and $\mathbf{C}_{1\leftarrow}$ respectively.

$$\mathcal{L}_{recon} = \mathbb{E}\left[\left\|\widehat{\mathbf{X}}_{1\rightarrow 1} - \mathbf{X}_1\right\|_2^2\right] \tag{2.1}$$

To avoid getting over-smoothed outputs as reported for the VAE model for VC proposed by Kameoka et al. (2018), an additional initial self-reconstruction loss term is added. The decoder D (module (c) in Figure 2.3) uses a residual network to build the finer details of the output. The usual self-reconstruction loss uses the decoder's output, but this same loss is also used on the rough reconstruction obtained before the residual network (2.2). Adding this term was reported to empirically improve the convergence in Qian et al. (2019).

$$\mathcal{L}_{recon_0} = \mathbb{E}\left[\left\|\widetilde{\mathbf{X}}_{1\rightarrow 1} - \mathbf{X}_1\right\|_2^2\right] \tag{2.2}$$

Alongside the previous reconstruction losses, an additional content code reconstruction error is added. The input $\mathbf{X}_1$ is transformed so that the codes $(\mathbf{C}_{1\rightarrow}, \mathbf{C}_{1\leftarrow}) := \mathbb{E}_c(\mathbf{X}_1)$ don't contain speaker information. As aforementioned, this is achieved by providing the uncompressed speaker information to the decoder. Speaker-independent codes will allegedly be achieved if the encoder compresses the signal enough, and the decoder is still able to obtain a faithful reconstruction. The content encoder applies an information bottleneck that both reduces the dimension along the mel-scale and downsamples the signal in the temporal dimension (modules (e,f) in Figure 2.3).

Note that if speaker disentanglement is achieved, the codes of the reconstructed signal $E_c\left(\widehat{\mathbf{X}}_{1\rightarrow 1}\right)$ should be the same as those of the original signal. The fact that the codes should match is because $\mathbf{X}_1$ and $\widehat{\mathbf{X}}_{1\to1}$ should only differ on the speaker information, which is not represented on the codes. The code consistency is enforced by adding the expected ℓ_1 norm between the codes from the original and reconstructed signal (2.3).

$$\mathcal{L}_{content} = \mathbb{E}\left[\left\|E_c\left(\widehat{\mathbf{X}}_{1\to1}\right) - E_c(\mathbf{X}_1)\right\|_1\right] \tag{2.3}$$

The AE, which includes the content encoder E_c and the decoder D, is then trained with the objective function in (2.4), where $\lambda = \mu = 1$ in Qian et al. (2019).

$$\min_{E_c,D} \{\mathcal{L}_{recon} + \mu\mathcal{L}_{recon_0} + \lambda\mathcal{L}_{content}\} \tag{2.4}$$

The output of the AE is mapped to raw speech by first upsampling the spectrogram to match the original speech waveform's sampling rate and then applying WaveNet conditioned on the upsampled spectrogram (van den Oord et al., 2016).

For all the qualities that AutoVC offers, this model was chosen as a starting point for this project.

2.1.2 CycleVAE

Tobing et al. (2019) proposed CycleVAE, a system that performs VC with non-parallel data by means of a compression-decompression scheme. Differing from AutoVC, the information bottleneck of CycleVAE is modeled with a VAE and not a plain AE, which means that the latents are enforced to follow a known distribution. CycleVAE learns one-to-one mappings, so a different model for each pair of source and target speakers is needed.

The breakthrough of Tobing et al. (2019) is that they propose incorporating VC during training. Performing VC during the training procedure helps overcome the performance degradation that comes from the impossibility to evaluate the converted speech. Having no parallel data means that there is no target signal for the converted speech, so the usual approach is to train on self-reconstruction. Even if this is the case, VC is incorporated in the CycleVAE training flow by performing more than one conversion and only evaluating those outputs whose target speaker is the original speaker. Each of such conversions is denoted as cycle and includes the computation of both the reconstructed (target speaker is the source speaker) and converted spectra (target speaker is not the source speaker).

The CycleVAE flow is depicted in Figure 2.4. Note that not only the source but also a target speaker is used during training in this case, and the reconstruction loss is computed when the converted speech at the end of one cycle corresponds to the source speaker. CycleVAE is then trained, minimizing the sum of all the reconstruction losses for each cycle.

CycleVAE uses spectral envelope parameters and excitation features as input. In particular, the first 35 Mel-Frequency Cepstral Coefficients (MFCCs) are used as spectral envelope parameters. For the excitation features, log-scale pitch contour, a voiced/unvoiced flag, and aperiodicity coding coefficients are used. The former parameters are computed using WORLD (MORISE et al., 2016).

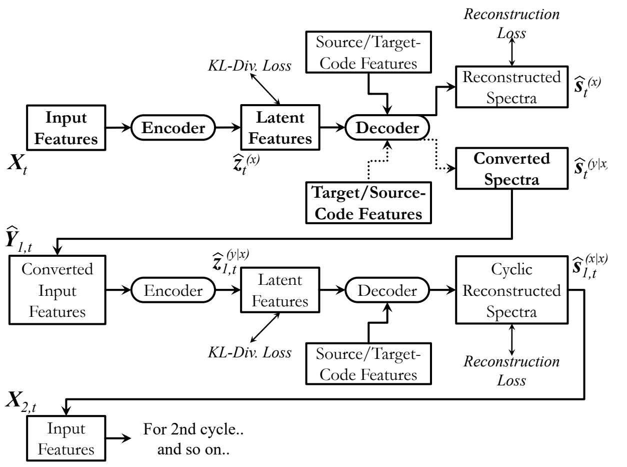


Figure 2.4: CycleVAE (Tobing et al., 2019) flow. The conventional VAE/glsae-based flow correspond to the first cycle (upper-part). Each encoder and decoder have the same parameters for all cycles.

Let f_0 be the random variable representing the instantaneous pitch or fundamental frequency. Let μ_{src} and σ_{src} be the mean and standard deviation of this quantity amongst all the speech signals of the source speaker. In order to preserve the intonation, which is determined by the pitch contour, the source speaker pitch is used. However, to resemble the characteristics of the target speaker, the estimated log-scaled pitch of the target speaker $\widehat{\log f_{0,trg}}$ obtained with (2.5) is used to condition the raw speech generator.

$$\widehat{\log f_{0,trg}} = \mu_{trg} + \frac{\sigma_{trg}}{\sigma_{src}}(\log f_{0,src} + \mu_{src}) \tag{2.5}$$

CycleVAE achieves fast conversions in its variant using Parallel WaveGAN (Yamamoto et al., 2019) as a raw speech generator[1].

2.1.3 VQ-VAE

Vector Quantization (VQ)-VAE was introduced in the paper entitled Discrete Neural Representation Learning by van den Oord et al. (2017). The main contribution of this work was the incorporation of VQ in the latent representation obtained by the encoder of a VAE, which gives discrete latents. The encoder of the VAE generates real latents that are then substituted by one of the K vectors in the codebook. In particular, the real latent is substituted with the codebook vector that is closer in the ℓ_2 norm sense.

The quantization operator is non-differentiable since it is a piece-wise constant function. To jointly train the encoder and decoder networks, van den Oord et al. (2017) use the straight-through estimator by Bengio et al. (2013), which consists in copying the gradients from the quantized latents to those before quantization. The codebook elements are updated using the VQ objective, which uses the ℓ_2 error to move the codebook elements towards the encoder outputs.

[1]An implementation of CycleVAE+Parallel WaveGAN used as a baseline for the Voice Conversion Challenge 2020 can be found in the following GitHub repository: `https://github.com/bigpon/vcc20_baseline_cyclevae`.

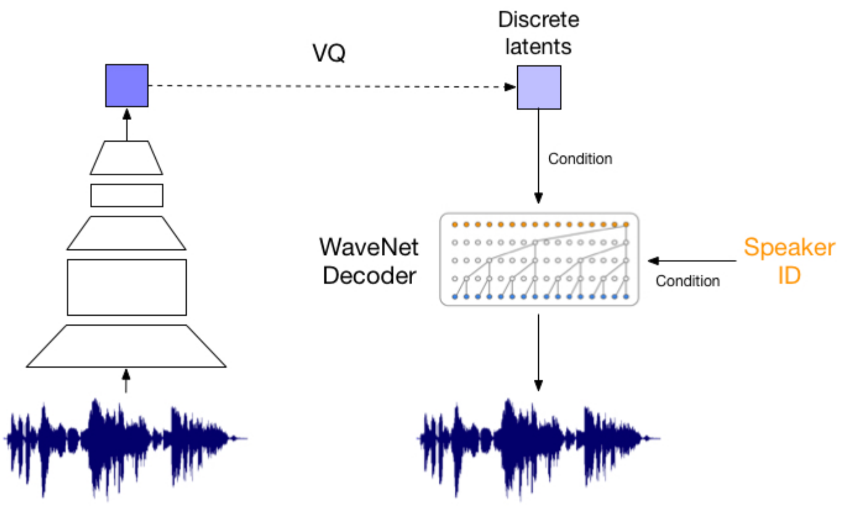


Figure 2.5: Schema of VQ-VAE in the configuration to perform voice-style transfer. Image taken from `https://avdnoord.github.io/homepage/vqvae/`.

VQ-VAE has been successfully applied to speech, images and videos (van den Oord et al., 2017). Its configuration to perform speech style transfer is shown in Figure 2.5. Note that, in this case, the decoder network is a WaveNet (van den Oord et al., 2016). The speaker disentanglement of the latents is achieved by conditioning such a decoder on both the latents and a speaker embedding. In particular, one-hot embeddings are used to represent the speaker's identity. This system is trained in reconstruction so that non-parallel data can be used.

2.2 Mapping from speaker-independent features

Given that VC requires the factorization of speech into linguistic and non-linguistic information, one natural approach is to use representations that lack speaker information. Such representations are then fed into a system along with the information of the target speaker.

Sun et al. (2016) proposed the use of Phonetic PosterioGrams (PPGs) over senones. PPGs are a two-dimensional feature representing the posterior probability of each phonetic class for each temporal frame. The phonetic class may be words, phones, or senones, which are context-dependent phonetic units.

2.2.1 ASR+TTS

Another approach that yields high-quality conversions is first transcribing the text using an Automatic Speech Recognition (ASR) system and then using a TTS model conditioned on the target speaker and the transcription. This approach's success comes from the fact that the two systems have been widely studied, and there are already several solutions at the user level in the form of virtual assistants. Some examples of these systems are Apple's Siri, Amazon's Alexa, or Google's Assistant. Such a system recognizes a command uttered by the user by using an ASR system. The transcription is then processed, and the query is answered by the virtual assistant using a TTS system.

A recipe to perform VC using the concatenation of ASR and TTS systems can be found in the open-source End-to-end Speech Processing Toolkit (ESPnet) (Hayashi et al., 2019; Inaguma et al., 2020; Watanabe et al., 2018). This system in particular is used as a baseline for the 2020 VC challenge.

The previous approaches use a pre-trained model to compute the speaker-independent representation. Such a model is not explicitly trained for the VC task, and its parameters are not modified to avoid the filtering of speaker information. Even if the ASR approach offers in general outstanding results, in the case of using text as an intermediate representation, the temporal information is completely lost. Moreover, most of the speaker-independent features also lack para-linguistical information such as the intonation, which can potentially lead to conversions having different meanings in some cases.

2.2.2 Adversarial training

Another approach to achieve speaker-independent representations while preserving para-linguistical and timing information is to extract latent variables from speech and explicitly enforce them to be speaker-independent. Enforcing speaker independence can be done with an adversarial setting, where a classifier is trained to predict the speaker from the latent representation. The loss from such classifiers can then be used to learn the mapping from the input signal to the latent space (Barbany, 2018, Section 3.2.2.). With this method, the representation can be better suited for the task of VC by jointly training the feature extractor and the conversion modules.

Chou et al. (2018) proposed such an adversarial setting for non-parallel VC. This work was one of the first successful implementations of a GAN-based model working with multiple speakers. Previous research in GANs for VC was mainly based in Zhu et al. (2017), which required one pair of generator and discriminator for each target speaker.

In this work, the input waveform is mapped to a latent representation with an encoder. This latter is then reconstructed with a decoder. This decoder uses both the latent representation computed with the encoder and the target speaker identity. The model is trained on reconstruction, and a speaker classification enforces the latent representation to be speaker-independent. By training an additional speaker classifier, Chou et al. (2018) showed that the adversarial classifier was needed to enforce speaker-independent latent representations for their proposed architecture.

With this approach, the obtained STFTs of the conversions tend to be blurry and have artifacts. To overcome this problem, Chou et al. (2018) proposed a second training stage where the generator of a GAN is trained to build the fine details of the spectrum obtained with the previous decoder.

MelGAN-VC is a model for non-parallel VC uniquely based on GANs with mel-spectrograms as input proposed by Pasini (2019). MelGAN-VC works with arbitrarily length audios by splitting and concatenating spectrograms but needs one pair of generator and discriminator networks for each target speaker. This method uses the classical Griffin-Lim algorithm (Griffin and Jae Lim, 1984) for mel-spectrogram inversion instead of using a conditional generative model. Note that even if the name of the method is MelGAN-VC, this model has nothing to do with MelGAN from Kumar et al. (2019).

Kameoka et al. (2018) proposed StarGAN, a GAN-based framework for non-parallel many-to-many VC. StarGAN solves the problem of needing one pair of generator and discriminator by speaker that Pasini (2019) has by conditioning both networks with the speaker identity. The generator learns to map the input features to the target domain, and the discriminator network tries to predict whether the generated signal belongs to the target domain or not. An additional domain classifier is used to ensure that the generated samples have the desired attributes, e.g., are classified as if they belong to the target speaker.

2.3 Speech Synthesis

One common approach for voice conversion is to leverage existing generative models for the task of raw speech generation. Using such models reduces the problem of VC to the generation of conditioning signals used for generative models to create raw speech. The conditioning signal is a representation with lower-resolution than the raw speech. Such representation is usually chosen to be efficiently calculated from the raw speech signal and easy to model. However, this representation has to preserve enough information to allow faithful inversion. Some examples of speech conditioning signals are time-aligned linguistic features and mel-spectrograms.

The same two-stage process of audio modeling is widespread in the literature, not only in VC but also in TTS. For this latter, a first model converts text into a conditioning signal, which is then fed to a generative model that produces raw speech (Wang et al., 2017; Sotelo et al., 2017). In this section, three of the most popular generative models for raw audio synthesis used in the recent VC literature are presented.

2.3.1 WaveNet

WaveNet (van den Oord et al., 2016) is a generative model for raw audio waveform generation. The model is fully probabilistic and AR, with the predictive distribution for each audio sample conditioned on the previous samples. The joint probability of a waveform is factorized as a product of conditional probabilities, whose distribution is modeled by a stack of dilated causal convolutional layers (see Figure 2.6). Causality makes WaveNet an AR model, and the dilation allows increasing the receptive field by orders of magnitude without needing many layers or large filters that would greatly increase the computational cost.

In particular, WaveNet exponentially increases the dilation factor, and with 32 layers where each convolution has a kernel size of 2, 1024 samples can be encompassed. WaveNet has a receptive field of 240 ms, which translates to 3840 samples at 16 kHz, the frequency at which WaveNet runs. Nevertheless, this is not enough to model some long-term dependencies like prosody. Bad prosody modeling translates into unnatural speech, but this problem can be solved by conditioning the WaveNet on pitch contours predicted with an external model at a lower frequency. In particular, WaveNet uses a prosody conditioning signal at 200 Hz. The same problem with the modeling of long-term dependencies appears in this project and is discussed in Section 4.2.

Training a WaveNet is fast, given that the CNN architecture allows parallelizing the prediction of the sample values at all time steps by using the ground-truth sample values to predict a new sample. However, during inference, each input sample must be drawn from the output distribution before it can be fed as input for the next sample's prediction. Therefore, WaveNet is poorly suited to parallel processing and hard to deploy in a real-time production setting.

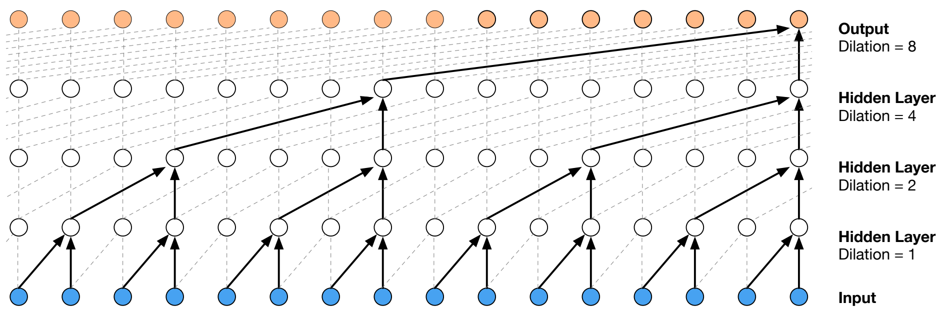


Figure 2.6: Stack of dilated causal convolutional layers from van den Oord et al. (2016).

2.3.2 MelGAN

MelGAN (Kumar et al., 2019) is a non-AR fully convolutional model for raw speech generation. Non-AR models are usually orders of magnitude faster than AR models. This is due to the fact that speech generation can be highly parallelized in the non-AR case, which is specially suited for GPUs and TPUs.

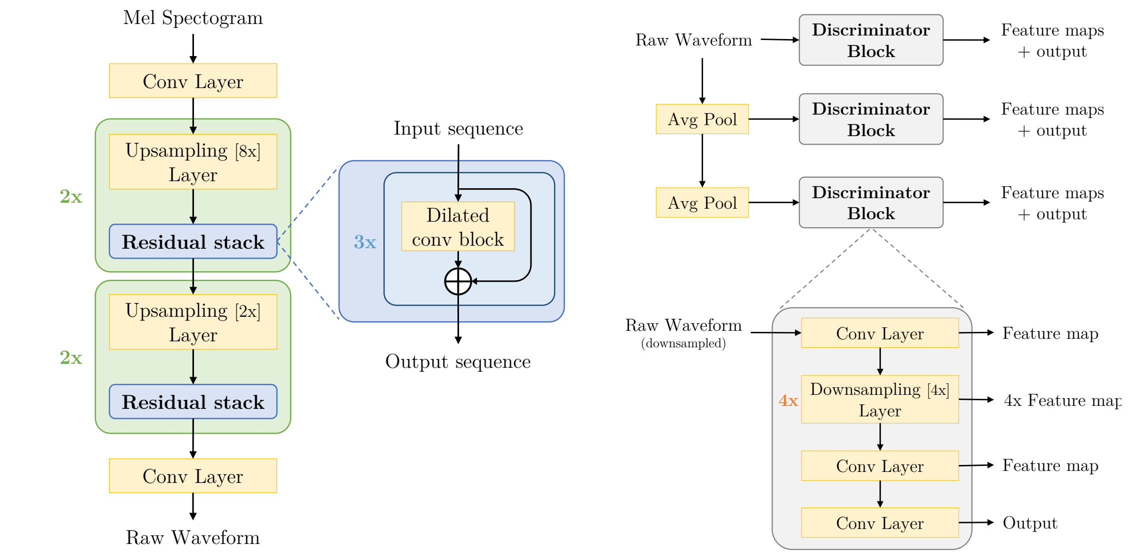
MelGAN learns the distribution of raw speech conditioned on log-scale mel-spectrograms through the two-player minimax game of a GAN (Goodfellow et al., 2014). Unlike regular GANs, MelGAN uses multiple discriminators that run at different rates, which fits the hierarchical structure of long and short time dependencies in speech. In particular, MelGAN uses three discriminators D_1, D_2 and D_3. D_1 operates on raw audio, D_2 on audio downsampled by a factor of 2, and D_3 on audio downsampled by a factor 4. The downsampling in all cases is performed using strided average pooling with kernel size 4. Several multi-scale discriminators introduce the inductive bias of the long and short term dependencies of speech. Kumar et al. (2019) shows that this multi-level structure is necessary because using the single discriminator used in traditional GANs introduces metallic audio.

Figure 2.7 depicts the MelGAN model architecture. Note that the generator is composed of dilated convolution, which again gives an inductive bias of the long-range correlation of speech signals as in van den Oord et al. (2016). The dilation factor is chosen to grow exponentially with the kernel size to avoid checkerboard artifacts, which in audio translates to high-frequency hissing noises.

Each discriminator D_k is trained using the Hinge loss version of the GAN objective (2.6), where $\mathbf{x}$ is the raw waveform and $\mathbf{s}$ the conditioning signal. The discriminators are window-based, which means that they learn to distinguish the distribution of real and fake audio chunks, not entire audio sequences. Such discriminators can be applied to audio sequences of variable length (Kumar et al., 2019).

$$\min_{D_k}\{\mathbb{E}_{\mathbf{x}}[\min(0, 1 - D_k(\mathbf{x}))] + \mathbb{E}_{\mathbf{s}}[\min(0, 1 + D_k(G(\mathbf{s})))]\} \; \forall k \in [3] \tag{2.6}$$

Note that in the above equations, the generator does not have a global noise input as traditional GANs do, even though the mel-spectrogram inversion is an ill-posed problem. Having a deterministic generative model is counterintuitive because there is not a unique mapping between the mel-spectrogram and the raw temporal signal. After all, the former is a lossy representation



(a) Generator (b) Discriminator

Figure 2.7: MelGAN (Kumar et al., 2019) model architecture.

of the latter. However, Kumar et al. (2019) hypothesizes that the conditioning imposed by the mel-spectrogram is very strong, and thus noise input is not essential.

Regarding the training of the generator, two losses are used: the usual discriminator's signal of GAN training (2.7), but also an additional feature matching loss (2.8). This latter is similar to a perceptual loss (Gatys et al., 2015; Johnson et al., 2016), and differs from other regularizers used in conditional GANs that try to match the input signal. This loss aims at matching each i−th feature map in every of the T layers of each discriminator for both real and generated speech.

The ℓ_1 norm on the difference between the generated output and the original raw waveform was reported to introduce sound artifacts in Kumar et al. (2019). On the other hand, the feature matching loss offers high-level descriptors that better model the nature of real speech signals.

$$\mathcal{L}_D(G, D_k) = \mathbb{E}_{\mathbf{s}}\left[-\sum_{k\in[3]} D_k(G(\mathbf{s}))\right] \tag{2.7}$$

$$\mathcal{L}_{FM}(G, D_k) = \mathbb{E}_{\mathbf{x}}\left[\sum_{i=1}^{T} \frac{1}{N_i}\left\| D_k^{(i)}(\mathbf{x}) - D_k^{(i)}(G(\mathbf{s}))\right\|_1 \middle| \ \mathbf{s} \text{ conditioning extracted from } \mathbf{x}\right] \tag{2.8}$$

The generator network is then trained with the objective function in (2.9) with $\lambda = 10$ in Kumar et al. (2019).

$$\min_G\{\mathcal{L}_D + \lambda\mathcal{L}_{FM}\} \tag{2.9}$$

MelGAN is a high-speed conditional generative model that offers comparable quality to some slower AR models, such as WaveNet (van den Oord et al., 2016). The downside of this model is that it assumes that the conditioning signal is temporally aligned to the input. So even if the generative model could potentially be conditioned on other signals than the mel-spectrogram, the length of the desired raw waveform has to be a multiple of the length of the conditioning signal.

2.3.3 Parallel WaveGAN

Parallel WaveGAN (Yamamoto et al., 2019) is yet another popular generative model for audio synthesis consisting of a non-AR model based on GAN. The generator network is inspired by WaveNet (van den Oord et al., 2016), but uses non-causal convolutions and uses random Gaussian noise as input. Therefore, the predicted samples are not fed back to the generator, and the speech generation process can be parallelized.

As it was the case for MelGAN, Parallel WaveGAN also uses an additional loss term alongside the adversarial loss given by the discriminator. The approach, in this case, is not to apply the ℓ_1 norm to the raw waveform nor to use feature matching as in MelGAN. Instead, ℓ_1 and Frobenius norms are used on the STFT magnitudes of the signal at different resolutions. The use of losses at several resolutions incorporates the inductive bias of the hierarchical structure of speech while it avoids forcing a match on the unsupervised representations given by the feature maps of the discriminators in MelGAN.

2.4 Voice Conversion Challenge

The Voice Conversion Challenge is a biannual workshop at INTERSPEECH, the world's largest technical conference on speech processing and applications. The first edition of the VC Challenge was held in 2016 and was aimed at evaluating and understanding different VC. The VC Challenge aims at improving VC technology, that is, without focusing on any particular application, using a common dataset, metrics, and baseline systems provided by the organizers. As mentioned in Chapter 1, one of the most significant difficulties of VC is the lack of objective measures. Subjective scores give an idea of the quality of a system, but the subjectivity makes comparing different approaches impossible solely based on these scores. The VC Challenge also evaluates the systems using subjective scores, but each system is rated by listeners from the same population. Since the compared systems are anonymized, and both the evaluators' population and the dataset are common, a faithful system comparison can be performed.

Another goal of the VC Challenge is to bring together different teams from both academia and industry to look at a common goal. In particular, each of the three challenges so far has introduced new unsolved problems, which make the VC problem harder.

The first edition of the Challenge (Toda et al., 2016) was centered in models trained on clean parallel data. The results of the Challenge set new baselines for the VC task and helped to share views about unsolved problems.

The second edition (Lorenzo-Trueba et al., 2018), included parallel but also non-parallel VC. The data also halved with respect to the past Challenge edition to force the systems to generalize with fewer data as shown in Table 2.1. This edition provided subjective comparisons but also introduced the evaluation of systems using spoofing performance (Kinnunen et al., 2018). The spoofing performance measured the vulnerability of speaker verification systems against samples

Challenge	Language	Training Utterances	Number of Speakers	Testing Utterances
VCC 2016	Monolingual	162 parallel	4 source, 4 target	54
VCC 2018	Monolingual	81 parallel	4 source, 4 target	35
	Monolingual	81 non-parallel	4 source, 4 target	35
VCC 2020	Monolingual	20 parallel, 50 non-parallel	4 source, 4 target	25
	Cross-lingual	70 non-parallel	4 source, 6 target	25

Table 2.1: Summary of VCC 2016, 2018, and 2020 adapted from Sisman et al. (2020).

obtained with VC systems and thus rated the matching of speech characteristics.

One of the teams that submitted samples to the 2018 Challenge (N10) vastly outperformed all the other systems in all the metrics. The results presented in the VC Challenge are anonymized. However, the organizers asked permission to the N10 authors and made an exception to describe this system, given its supremacy. The N10 system by Liu et al. (2018) achieves VC with a conversion function that converts the acoustic features extracted from the source speech using the STRAIGHT (Kawahara, 2006) vocoder. Such converted acoustic features are used to condition WaveNet (van den Oord et al., 2016), which generates raw speech.

In particular, the N10 system uses acoustic features, including MFCCs and pitch, and a speaker embedding. The conversion model includes a speaker-independent content feature extractor, mapping the former acoustic features to phonetic transcriptions. Such mapping is learned using hundreds of hours of recording with aligned phonetic transcriptions. The conversion function also includes a speaker-dependent feature predictor that converts the content features to the target speaker. Since the content feature extractor and the content predictor are trained separately, the N10 system can be used in parallel and non-parallel settings. The authors of this system also performed some manual checks and annotations of the VC Challenge data. Some of these manual checks include pitch extraction errors and the removal of speech segments with irregular phonation (Lorenzo-Trueba et al., 2018).

The third edition of the Challenge (VCC20) registered the highest participation amongst the previous editions, with more than 90 research groups around the globe enrolled. This edition was centered on non-parallel training, and two tasks were presented on this framework.

Similarly to the 2018 edition, one task was to perform VC within the same language. The second task, however, introduced a new problem termed as cross-lingual VC. In this setting, the source and target speakers speak different languages. Note that, following the speech style transfer and VC formulation, the source content is unaltered, so this task does not include translation of the linguistic content.

For the 2020 edition of the Challenge, the organizers provided speech samples and implementations of two baseline systems. One of these systems was an implementation of CycleVAE (Tobing et al., 2019) with Parallel WaveGAN (Yamamoto et al., 2019) as a conditional generative model (see Section 2.1.2). The second baseline was a concatenation of an ASR system and a TTS system (see Section 2.2.1) using ESPNET (Hayashi et al., 2019; Inaguma et al., 2020; Watanabe et al., 2018) and Parallel WaveGAN (Yamamoto et al., 2019). Samples from the N10 system from the 2018 edition were also provided for comparison.

2.5 Applications

VC can be used to modify a speech signal so that it sounds as spoken by a target speaker. The ability to clone someone's voice can be used to customize audiobooks, dubbing in the movie industry, and clone voices of historical persons. This internship was driven by the application of VC to customizing avatar voices, which could be used to offer speech anonymization, which preserves the privacy of the speaker and can potentially be a desirable feature in environments such as online gaming. This could also avoid gender discrimination in this environment.

Some of the non-obvious applications of VC include aids for vocally impaired people such as patients of dysarthria (Kumar and Kumar, 2016) and laryngectomees (Doi et al., 2014). These methods consist of enhancing the speech uttered by people with some disease affecting speech by improving its intelligibility. They can be formulated as a speech transfer problem, where the content is the linguistic information as in VC. However, in this case, the target style is the patient's voice without any vocal impairment.

Another related application is silent speech interfaces (Toda et al., 2012). In this case, there is no input speech signal, but the aim is to generate speech based on other signals such as ultrasound images of the tongue and lip movements or the electromyography of the larynx and the speech articulator muscles. This technique is thus also suited for vocally impaired patients but can also be used as electronic lip reading. This problem can also be formulated in the framework of style transfer, but with signals of different nature.

VC can also be applied to voice changing for expressive speech (Turk and Schroder, 2010), where instead of changing the speaker of a speech signal, other para-linguistical factors such as the intonation and the stress are modified. Other uses of VC are accent conversion for computer-assisted learning (Felps et al., 2009) and vocal effects for singers (Villavicencio and Bonada, 2010).

VC can also have malicious applications such as the fooling of speaker verification systems. One of the main goals of VC is to match the speaker characteristics of the target speaker, and thus VC systems can potentially match the biometrics derived from speech. Given that subjective scores are not directly related to spoofing, Kinnunen et al. (2018) proposes an assessment of VC systems with spoofing measures computed with biometric speech systems. The advances in VC are also relevant to build robust verification systems and are used in spoofing databases as adversarial examples (Wu et al., 2017).

3 Methodology

As mentioned in Chapter 1, a common approach for audio and speech applications is to use the STFT, or one of its variants such as the mel-spectrogram, as input. In this chapter, we discuss a method to implement this transformation with CNNs. The advantage is that the initial weights give the usual transformation, but such weights can be learned. Learning the transformation from waveforms can potentially find better representations suited for each task and recover information not included in the mel-spectrogram due to its lossy nature.

This chapter also describes the two main Speech Style Transfer approaches that were followed through all this project. Both systems are trained using non-parallel data and perform many-to-many VC. The sole input of both systems is raw speech and speaker identifiers, so no annotations or other kinds of additional information are required.

On the one hand, we propose FastVC, an AE-based framework that disentangles the linguistic and speaker's information using a compression-decompression scheme. This method lies in the class of conversion functions described in Section 2.1.

On the other hand, we explore a method based on mapping speaker-independent features to raw speech. This family of methods is described in Section 2.2 and alleviates the conversion model from separating the speaker's information by using features that lack this content. Using speaker-independent features is a more straightforward approach, but some representations may not be speaker-independent or missing crucial linguistic information.

In the last section of this chapter, the experimental setup of all the experiments is described.

3.1 Fourier initialization

The implementation of learnable versions of digital signal processing components was already studied in (Engel et al., 2020). Implementing speech processing techniques with learnable models, in general, endows such models with powerful inductive biases about the structure of speech signals. As discussed in Chapter 1, these inductive biases are, in most cases, beneficial and allow for better results. Moreover, some approaches also reduce the number of free parameters.

One example of an implementation of a typical signal processing filter with neural networks is SincNet. This model was proposed by Ravanelli and Bengio (2018) and restricts the convolutional filters to implement band-pass filters with learnable cutoff frequencies. In other words, each filter only has two learnable parameters: the low and high frequencies determining the ideal band-pass filter. SincNet is thus one case of inductive bias that reduces the number of needed weights.

Tancik et al. (2020) proposed Fourier feature mappings, where the output is restricted to be the pair of sine and cosine of some higher-dimensional vector that is later fed to a neural network. This Fourier feature mapping is presented in (3.1), where $\mathbf{v}$ is an input point and $\mathbf{B}$ a random Gaussian matrix where each entry is drawn independently from a zero-mean normal distribution.

$$\gamma(\mathbf{v}) = [\cos(2\pi\mathbf{B}\mathbf{v}), \sin(2\pi\mathbf{B}\mathbf{v})]^T \tag{3.1}$$

Implementing the function mapping raw speech to mel-spectrograms with CNNs does not restrict the neural network's expressive power. Instead, this allows training with raw waveforms as input with any of the vast family of deep learning models working with mel-spectrograms.

The proposed initialization sets the weights of the transformation network to the Fourier eigenvectors and the mel filter-bank coefficients. This initialization aims at providing the same performance both when using mel-spectrograms as input and at the beginning of plugging the proposed model and using waveforms as input. Switching from deterministic mel-spectrogram computation to a learnable transformation that provides the same results allows training on top of an already working model. When choosing an adequate objective function, this could be used to improve the model performance further.

The STFT is a powerful general-purpose audio processing tool (Smith, 2011). The purpose of the mel-scale is to represent the perceived difference between frequencies. Humans can easily differentiate low frequencies that differ e.g., 100 Hz, but this same distance is hardly perceived in high frequencies.

For a window $\mathbf{w} \in \mathbf{Z}^N$ and hop length h, the STFT of a waveform $\mathbf{x}$ is defined as follows:

$$\mathbf{X}[m, \omega] := \sum_{k=0}^{N} \mathbf{x}[m \cdot h + k]\mathbf{w}[k]e^{-j\frac{2\pi\omega k}{N}} \tag{3.2}$$

The STFT is complex valued, but we can decompose it into its real and complex part, which can be written in the form of a 1-dimensional convolutional layer with stride h and kernel size N.

$$\mathbf{X}[m, \omega] = \sum_{k=0}^{N} \mathbf{x}[m \cdot h + k]\mathbf{w}[k] \cos\left(\frac{2\pi\omega k}{N}\right) - j\sum_{k=0}^{N} x[m \cdot h + k]w[k] \sin\left(\frac{2\pi\omega k}{N}\right) \tag{3.3}$$

$$\mathbf{X} = \text{Conv1D}(\mathbf{x}; \boldsymbol{\theta}_{0,\text{Real}}) - j\text{Conv1D}(\mathbf{x}; \boldsymbol{\theta}_{0,\text{Imag}}) \tag{3.4}$$

where $\boldsymbol{\theta}_{0,\cdot}$ contains the initial weight matrix and bias vector defining the 1-dimensional convolution. In particular, the bias is initialized to zero, and the weight comprises the window coefficients and the real and imaginary parts of the Fourier eigenfunctions.

The mel-spectrogram $\mathbf{S}$ can be written as.

$$\mathbf{S} := \mathbf{M}|\mathbf{X}| = \mathbf{M}\sqrt{\text{Conv1D}(x; \theta_{0,\text{Real}})^2 + \text{Conv1D}(x; \theta_{0,\text{Imag}})^2} \tag{3.5}$$

where $\mathbf{M}$ is the filterbank matrix to combine Fourier Transform bins into mel-frequency bins. Note that (3.5) can be also expressed with a 1-dimensional CNN with kernel size 1. This latter basically consists in a matrix multiplication followed by the addition of some bias, which is initialized to zero.

3.2 FastVC

FastVC is an end-to-end model that performs fast many-to-many cross-lingual VC and is trained using non-parallel data. This system is mainly based on AutoVC by Qian et al. (2019), which is described in Section 2.1.1. FastVC belongs to the family of models described in Section 2.1, which perform VC by learning a mapping between the speech features of the source speech and those of the converted speech. This latter has the same linguistic information as the source speech but different speaker information. In particular, FastVC learns this mapping with an AE framework that is trained on the reconstruction of mel-spectrograms. The FastVC model architecture is depicted in Figure 3.1.

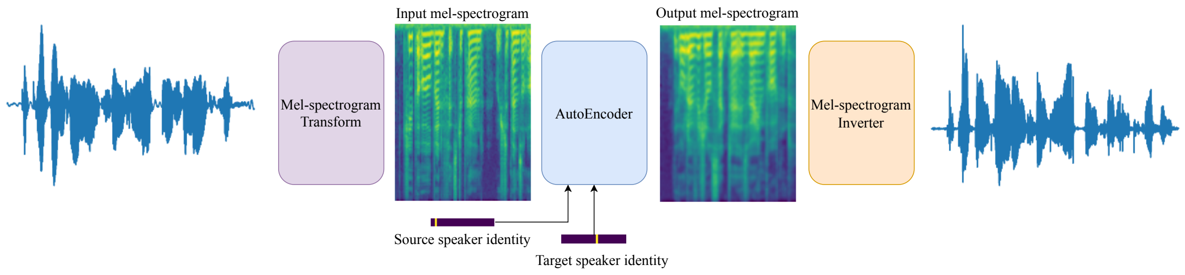


Figure 3.1: FastVC model architecture during conversion mode. During training, both the input and output speech and the source and target speaker embeddings are the same.

Equally to AutoVC, FastVC uses log-scale mel-spectrograms as inputs. However, the `Mel-spectrogram Transform` module in Figure 3.1 is a CNN-based learnable module and not a fixed transformation as in AutoVC. This module can be trained and is initialized to provide exact mel-spectrograms using the techniques described in Section 3.1.

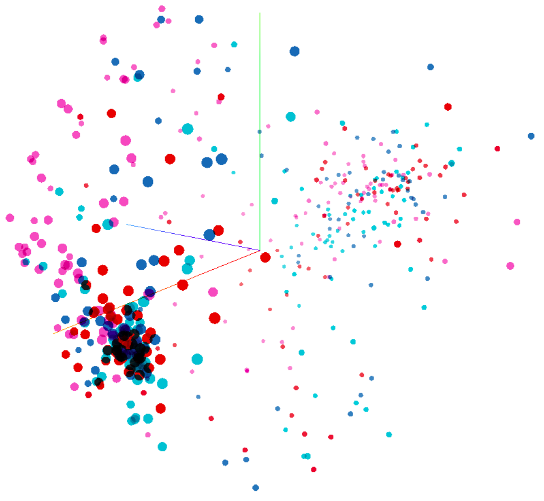
To obtain the inputs to the `AutoEncoder` module in Figure 3.1, the spectrogram of the raw speech is passed through an 80-channel mel-scale filter-bank. As shown in (3.5), the filter-bank matrix combines Fourier Transform bins into mel-frequency bins. In particular, AutoVC uses a filter-bank with minimum and maximum frequencies. By default, the mel-scale transformation considers all the frequencies, ranging from 0 to half the sampling rate. Setting a range of frequencies like in Qian et al. (2019) provides a higher resolution for such range. Having a higher resolution for the mid-range is consistent with the fact that the human ear is not very sensitive at low or high frequencies compared to the highest sensitivity attained in the $1-5$ kHz range. However, obviating high frequencies can result in loss of intelligibility, and not considering low frequencies can result in poor modeling of speech features such as the pitch. Given the previous justifications, which are in line with the transformation performed in Kumar et al. (2019), no minimum and maximum frequencies were set in the mel-scale transformation. Finally, a log dynamic range compression is applied to the resulting mel-spectrogram.

Before computing the mel-spectrogram, Qian et al. (2019) applies reflection padding to the input waveform. FastVC follows this same approach but using the same amount of padding as in Kumar et al. (2019), which differs from the one used in AutoVC. Qian et al. (2019) then apply signal pre-processing techniques to the padded waveform, including the filtering with a Butterworth high pass filter (Butterworth, 1930) and the addition of noise. The Butterworth filter is a high-pass filter with a very flat response before the cutoff frequency by design and has a quick roll-off around such frequency value. This filter is applied to remove the high-frequency noise of the original signal, and then synthetic noise is added for robustness purposes. These techniques were not incorporated in FastVC, which is again in line with the design choices of

Kumar et al. (2019).

The theoretical guarantees justifying the VC capabilities of AutoVC hold under the assumption that the speaker embeddings of different utterances of the same speaker are the same, and those from different speakers are distinct. Even if this is consistent with the GE2E loss used to train the speaker encoder, the speaker embeddings differ from one speaker to another. In fact, those speaker embeddings do not exhibit the assumed properties, as shown in Figure 3.2.

Qian et al. (2019) solve this issue by using the average speaker embedding of 10 two-second speech signals of the same speaker. However, given that the speaker embeddings do not cluster, this solution may not be the best. Instead, FastVC uses one-hot encoded speaker embeddings, which satisfy the former assumption and suffice for conventional many-to-many VC. Using one-hot embeddings means that the speaker conditioning signal is not extracted from data, and thus it is exactly the same for all the audios of each speaker. One-hot encoded speaker embeddings are used in van den Oord et al. (2017), which uses a similar framework for VC.


(a) Representation using Principal Component Analysis (PCA). Before applying the dimensionality reduction, the data was normalized by shifting each point by the centroid and making it unit norm. This preprocessing is performed by default in Smilkov et al. (2016).

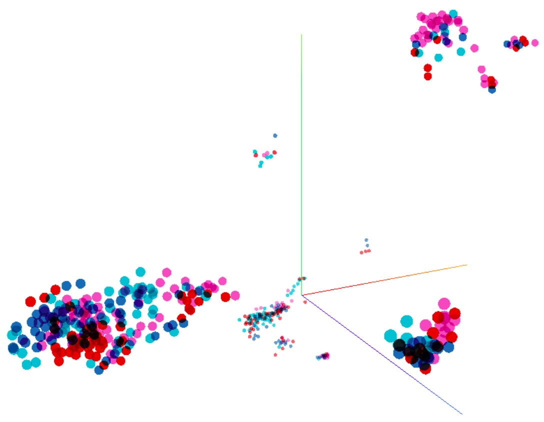

(b) Representation in the iteration 2000 of unsupervised t-Distributed Stochastic Neighbor Embedding (t-SNE) with perplexity 30, learning rate 10.

Figure 3.2: 3-dimensional representations of the 256-dimensional speaker embeddings obtained with the style encoder of AutoVC using the architecture and weights provided by Qian et al. (2019). In particular, this embeddings represent 480 utterances sampled uniformly from the female speakers `p225,p228` (represented with light and dark blue, respectively) and the male speakers `p226,p227` of the VCTK dataset (Veaux et al., 2016) (represented with red and pink respectively). This representations have been obtained using the embedding projector tool developed by Smilkov et al. (2016).

The larger overlap between AutoVC and FastVC takes place in the `AutoEncoder` module in Figure 3.1. Given that this implements the conversion function and thus is the core for VC, this module is depicted with more detail in Figure 3.3.

The encoder network learns to factor out speaker information in the latent representations in an unsupervised way. This behavior arises naturally because the decoder gets the source speaker identity for free during training, so the limited bandwidth of the latents is used to represent the

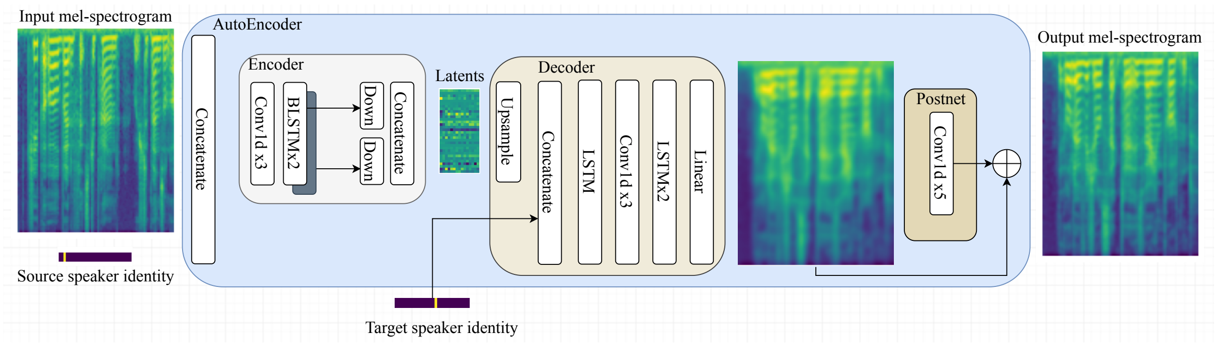


Figure 3.3: Diagram of the AE module for FastVC. The `AutoEncoder` comprises the `Encoder` and the `Decoder`, but also the `PostNet`. The `PostNet` builds the finer details of the spectrogram, which is excessively smooth before this module.

speaker-independent information.

Chorowski et al. (2019) claimed that the latent representations found by AEs do not factor the speaker out. Instead, their findings showed that to achieve speaker disentanglement, a VAE, which produces distributions over latents instead of deterministic encodings, or its VQ version, was required. However, Heck et al. (2016) showed that conditional AEs with speaker-dependent encoders effectively achieve the desired speaker-independent latent representations. This is consistent with the fact that the frequency representation of the speech sounds depend on the vocal tract structure of the speaker.

In FastVC, both the encoder and the decoder are conditioned on the speaker identity of the source and target speaker, respectively. This speaker identity is concatenated with the other input signal at every time step. For a mel-spectrogram of N channels and T temporal bins, the resulting input has shape $(N + S) \times T$, where S is the dimension of the speaker embedding. Note that this corresponds to the number of distinct speakers in the dataset when using one-hot embeddings.

One of the most crucial design choices in implementing a conversion function for VC learned on speech reconstruction is choosing adequate information bottlenecks. FastVC also uses dimensionality reduction in the frequency dimension and temporal downsampling. The former is very easy to implement since it only needs to reduce the number of channels in the encoder and to augment it back to match the input value. Downsampling a signal is also trivial when no bandwidth reduction is applied since it only consists of discarding samples. In this case, no anti-aliasing filter is applied, but the original signal is directly decimated. There are many methods to perform upsampling, and it can be critical in some cases. FastVC takes the same approach as Qian et al. (2019), which consists of performing upsampling with a causal variant of the nearest neighbor interpolation technique.

Chorowski et al. (2019) proposed the use of the time-jitter regularization. This approach helps to model the slowly-changing phonetic content by avoiding the use of latent vectors as individual units. The time-jitter regularization consists of replacing each latent vector with either one or both of its neighbors. The temporal upsampling approach that FastVC follows can be interpreted as a causal version of the time-jitter regularization proposed in Chorowski et al. (2019), with the time jitter as a hyper-parameter that corresponds to the downsampling factor.

The information bottleneck introduces two pivotal hyperparameters for the speaker disentanglement; the latents' dimension and the downsampling factor. As aforementioned in Section 2.1, these values should provide enough compression so that speaker information cannot be stored in the latent representation. At the same time, the compressed signal should contain enough information so that, along with the target speaker identity, a faithful reconstruction is possible.

In particular, FastVC doubles the temporal downsampling factor with respect to Qian et al. (2019). This design choice achieves speaker-independent phoneme-like latents that solve the pitch inconsistency problems of AutoVC reported in Qian et al. (2020). See Section 4.2 for more details.

FastVC generates speech with a sampling rate of 22050 Hz, which makes AR models unsuitable, especially if fast conversions are desired. AR models produce one sample at a time by conditioning the model on the past samples, so generating a waveform at this rate can take plenty of time. The generation is obviously slower for complex models requiring a vast number of operations to generate each sample.

Obtaining fast conversions was one of the desired specifications of the VC models developed in this work, so AR models were not adequate. In the original AutoVC paper, a WaveNet (van den Oord et al., 2016) conditioned on the log-scaled mel-spectrogram is used as a generative model for raw speech. This model is one of the most widely used generative models for raw speech synthesis due to its high speech quality but requires tens of billions of floating-point operations per second. This number of operations is too high for running in real-time on a CPU. In fact, this is even too high for a GPU, given that the model generates one sample at a time and cannot be parallelized. Overall, WaveNet requires several minutes to generate only one second of audio.

To achieve fast inference, FastVC resorts to using a non-AR generative model. This design choice is the main reason for the fast conversions obtained with this approach. Moreover, using a generative model circumvents the problem of generating raw speech with only the magnitude of the spectrogram, as already mentioned in Chapter 1. In particular, the mel-spectrogram inverter is chosen to be MelGAN by Kumar et al. (2019).

3.2.1 Training

GANs are notoriously difficult to train, with mode collapse and oscillations being a common problem (Liang et al., 2018). For this reason, the basic FastVC model uses the pre-trained weights for MelGAN provided by Kumar et al. (2019). The publicly available weights for MelGAN require a different conditioning signal than the WaveNet variant used in AutoVC. In particular, the differences in the required conditioning signal of the raw speech generative model are in the hop length parameter and the padding strategy. The former gives a conditioning signal of different temporal rates, and the latter modification has to be incorporated to exactly match the length of the input waveform. The public MelGAN weights are trained for 400k iterations on the LJ Speech dataset by Ito and Johnson (2017). This dataset is disjoint from the one used to train FastVC (see Section 3.4.1). However, Kumar et al. (2019) show that MelGAN can generalize to unseen speakers. In particular, the generalization test is performed on VCTK, one of the datasets used to train FastVC.

Even if the authors of AutoVC shared the pre-trained weights for this model, the AE module in FastVC is trained from scratch to match the conditioning signal required by the mel-spectrogram inverter. The coupling of the AE and the mel-spectrogram inverter is done by training the former instead of the latter because training MelGAN is significantly more complex than training a simple AE. The basic version of FastVC is obtained by only training the AE module using (2.4), which is the same loss used to train AutoVC. The initial rough mel-spectrogram reconstruction error (2.1) has the same weighting as the regular mel-spectrogram reconstruction error (2.1), which is at the same time the same as the content error (2.3). In other words, $\mu = 1$ and $\lambda = 1$ in (2.4). This configuration is trained using the ADAM optimizer by Kingma and Ba (2014) with the default PyTorch (Paszke et al., 2019) parameters, that is, a learning rate of 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.99$.

After training the AE of FastVC, the resulting reconstructed log-mel spectrogram suffers from over-smoothing (see Section 4.1). Since the generative model for raw speech synthesis is conditioned on original log-mel spectrograms, this potentially generates some artifacts. To allow the model to use information that may be not included in the mel-spectrogram or generate more efficient representations for the task of VC, FastVC also allows for the joint training of neighboring modules. That is, the input transformation can be learned at the same time as the weights of the AE. Also, the AE and the mel-spectrogram inverter can be jointly trained, and also real end-to-end training can be achieved. Additionally, the fine-tuning of the mel-spectrogram inverter weights is also possible.

FastVC is trained by phases when updating the weights of one of the subsets of modules of the former paragraph. The training by phases is done by fixing coherent intermediate representations and using pre-trained modules. Put differently, the basic module serves as a starting point, and the training that updates the parameters for other modules is considered fine-tuning. FastVC follows this approach to avoid the vast amount of data or time and computational resources that training an end-to-end system of these characteristics from scratch would require. During the training of other modules than the AE, the intermediate representations are relaxed. In other words, when the transform is learned, the autoencoder uses other inputs than a simple mel-spectrogram, and when MelGAN is trained, the conditional signal can also differ from mel-spectrograms.

In the case that the AE is the last trainable module, the same objective function used for the basic configuration is chosen, and the same training techniques are applied. When also training MelGAN, the discriminators' objective function is chosen to be (2.6) as in Kumar et al. (2019). To ensure that the linguistic information is captured, a cycle-consistency loss term is added. This loss term enforces the codes of the original and converted speech to be the same, i.e., it enforces VC to be idempotent. We speculate that this is enough to achieve quality speech that preserves the lexical content.

In particular, an additional regularization term is added to the generator objective (2.9) to consider the code consistency that is specific of the VC problem and not included in the loss function optimized in Kumar et al. (2019). The content loss (2.3) is weighted by $\lambda = 20$ and added to the total objective of the generator. The amount of regularization for the content loss is chosen to ensure that all the losses have the same order of magnitude, so the content loss is considered, and the weight updates decrease its value. For this setting, ADAM is also used as the optimization algorithm. In this case, however, with a learning rate of 10^{-4}, $\beta_1 = 0.5$, and $\beta_2 = 0.9$. These specific values are suggested in Daskalakis et al. (2017) to train GANs with ADAM, and also used in Kumar et al. (2019).

3.2.2 Variants

The AE module in FastVC in the basic configuration is trained to achieve a reconstruction with the minimum distortion in the ℓ_2-norm sense. Achieving a perfect reconstruction in an AE is burdensome, and in the case of FastVC, the reconstructed mel-spectrograms suffer from over-smoothing (see Section 4.1). Using a generative model for mel-spectrogram inversion can compensate for such imperfections, but it is worth studying methods that minimize the perceived artifacts given a value of distortion. The ℓ_2 norm on STFTs is arguably far from perceptual measures and could create artifacts. Using the mel-spectrogram already considers human perception, but some other metrics could be a better fit.

An approach to obtain a mel-spectrogram that corresponds to perceptually good speech was to try the power-law loss instead of the Mean Squared Error (MSE). The change in the reconstruction objective of the AE module was motivated by Kim and Stern (2010), which claimed that power-law loss correlates with the human perception of loudness. In particular, in the power-law experiment, FastVC used the variant of (3.6) with $\alpha = 0.5$ as described in Kadioglu et al. (2020). This same objective was used with the mel-spectrogram instead of the STFT.

$$\text{P-law}(x, y, \alpha) = \| \|\text{STFT}(x)\|_F^{\alpha} - \|\text{STFT}(y)\|_F^{\alpha} \|_1 \tag{3.6}$$

However, the use of this metric on the FastVC AE module resulted in non-converging weights. This is probably the case because the power-law does not only attain its minimum when $\text{STFT}(x) = \text{STFT}(y)$ as does the MSE. One sufficient condition for the minimum is to have spectrograms with the same singular values. Moreover, the choice of $\alpha < 1$ eliminates the desirable convexity of the norms.

Inspired by the cyclic training of Tobing et al. (2019), a variant of FastVC with VC in the training flow was also analyzed. However, instead of computing conversions and reconstructions at every cycle as CycleVAE (see Section 2.1.2), the approach was to perform VC two times. The source speaker's characteristics are changed by those of the target speaker in the first forward pass of FastVC. The AE is trained on reconstruction, so a second VC is performed with the same FastVC model to map the speaker characteristics of the previous output back to the source speaker. The objective function with cyclic training is very similar to (2.4) with two main differences. First of all, both the reconstruction and the rough reconstruction are the outputs of the second conversion. Secondly, the content loss is the average of the content losses of both forward passes. Unlike the previous approach, the cyclic training resulted in decreasing losses, but the results were perceptually worse than those obtained with the usual reconstruction training.

AEs find compact representations of the input signal that allow for its recovery. FastVC requires that these representations are speaker-independent, but this is not explicitly enforced. The fact that the encoder disentangles the speaker in an unsupervised fashion can be explained with the redundancy principle (Barlow, 1989). However, adversarial training of the latent representations as in Chou et al. (2018) could further disentangle the speaker's information and downplay the design choices of the information bottleneck.

In this variant, FastVC uses a speaker classifier that is an adaptation of the ZDiscriminator used in Mor et al. (2019), where this module is used to achieve class-independent latent representations in the Universal Music Translation Network. The minimax game here is for the encoder to seek class-independent latents and the classifier to classify them correctly. This network is trained with the cross-entropy loss on the speaker labels using the ADAM optimizer with a learning rate of 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.99$. The negative loss, termed as domain confusion loss in Mor et al. (2019), is added as a regularizer to the global objective with a weighting of $\eta = 0.1$.

Phonological Class	**List of phonemes**
Vocalic	/a/, /e/, /i/, /o/, /u/
Consonantal	/b/, /tʃ/, /d/, /f/, /g/, /x/, /k/, /l/, /ʎ/, /m/, /n/, /p/, /ɾ/, /r/, /s/, /t/
Back	/a/, /o/, /u/
Anterior	/e/, /i/
Open	/a/, /e/, /o/
u Close	/i/, /u/
Nasal	/m/, /n/
Stop	/p/, /b/, /t/, /k/, /g/, /tʃ/, /d/
Continuant	/f/, /b/, /tʃ/, /d/, /s/, /g/, /ʎ/, /x/
Lateral	/l/
Flap	/ɾ/
Voice	/a/, /e/, /i/, /o/, /u/, /b/, /d/, /l/, /m/, /n/, /r/, /g/, /ʎ/
Strident	/f/, /s/, /tʃ/
Labial	/m/, /p/, /b/, /f/
Dental	/t/, /d/
Velar	/k/, /g/, /x/
Pause	/sil/

Table 3.1: Classification of Spanish phonemes into phonological classes from Vásquez-Correa et al. (2019).

3.3 PhonetVC

PhonetVC is a model trained on non-parallel data that performs fast many-to-many VC. This model was conceived as a variant of FastVC. However, it is presented in a separate section since it belongs to a different family of methods (see classification in Chapter 2). The idea of PhonetVC is to avoid relying on the encoder network of the AE module in FastVC to disentangle the speaker's information. In FastVC, the speaker-independence of the latents is justified with the redundancy principle or the adversarial classifier. However, in these cases, the latent representation may still contain speaker information, as reported in Qian et al. (2020). Using speaker-independent inputs alleviates the VC from disentangling the speakers' information from the latent representations, but depends on the performance of pre-trained feature extractors. Moreover, these systems require, in general, large amounts of transcribed data, whose collection is very costly and time-consuming.

PhonetVC takes a similar approach as Sun et al. (2016) and solves this problem by using probabilities of phonetic classes as input to the decoder instead of latent representations. In particular, PhonetVC relies on Phonet, a toolkit to compute posterior probabilities of phonological classes by Vásquez-Correa et al. (2019). Phonological features are speaker-independent and directly related to which sound is being uttered. The motivation of Phonet comes from pathological speech processing, where phonological features are attractive because of their interpretability, which comes from its relation to the movements of the articulators in the vocal tract.

Phonet estimates posterior probabilities of the occurrence of different phonological classes using a bank of parallel bidirectional Recurrent Neural Networks (RNNs). The models proposed in Vásquez-Correa et al. (2019) are trained on Mexican Spanish annotated speech data and can detect the phonological classes with accuracy over 90%. The variant used in PhonetVC uses all the phonetical classes shown in Table 3.1, which results in an 18-channels input.

PhonetVC uses an estimation of the Phonological Log-Likelihood Ratio (PLLR) features instead of the latent representations found by the encoder of FastVC. The schema of PhonetVC is depicted in Figure 3.4. The decoder network takes its architecture from FastVC, and the chosen mel-spectrogram inverter is again MelGAN (Kumar et al., 2019). Given that the decoder's input is different from that in FastVC, this network is trained from scratch using the same objective as FastVC. The output mel-spectrogram is then fed to MelGAN.

Phonet works with waveforms sampled at 16 kHz. In order to use the latter, PhonetVC also generates speech at 16 kHz. As mentioned in Section 2.3.2, one of the downsides of MelGAN is that it assumes that the conditioning signal is temporally aligned to the input. Moreover, the MelGAN pre-trained weights provided by Kumar et al. (2019) are trained with audio at 22050 Hz and different windowing strategies. In particular, the hop length value used in Phonet is different from the default value in MelGAN, which forces the use of upsampling layers with different upsampling ratios in the MelGAN generator. The mel-spectrogram inverter is adapted to not use waveform padding before the representation in temporal bins to match the design choices of Phonet.

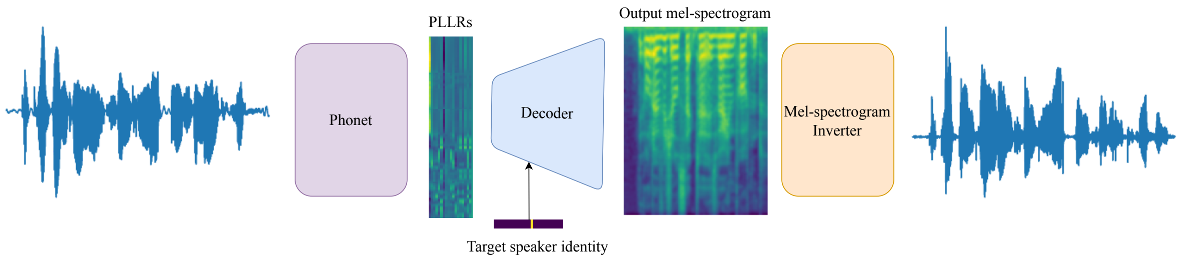


Figure 3.4: PhonetVC model architecture during conversion mode. During training, the target speaker embeddings represents the speaker uttering the input waveform.

3.4 Experimental setup

This section describes the setup for the experiments performed with PhonetVC, FastVC, and the latter variants.

3.4.1 Dataset

The main dataset used for this project is the Voice Cloning Toolkit (VCTK) described in Veaux et al. (2016). The VCTK dataset is chosen as the main dataset for its widespread use for the VC task (van den Oord et al., 2017; Chou et al., 2018; Qian et al., 2019, 2020). Moreover, MelGAN is shown to generalize to the speakers of this dataset even if not seen during training.

The VCTK dataset is a public speech dataset consisting of 44 hours of English speech and its transcriptions. There are a total of 109 speakers with various accents, and on average, each of them has around 400 utterances and 24 minutes of data. Even if there are some parallel utterances, the dataset is essentially non-parallel. The VCTK corpus speech was recorded in a hemi-anechoic chamber of the University of Edinburgh using an omnidirectional microphone. The speech was recorded with a sampling frequency of 96kHz sampling frequency and with a 24-bit resolution and then downsampled to 48 kHz and converted into 16 bits.

The model comparison of the VC Challenge is performed with samples generated using the dataset of the same Challenge. This dataset has ten speakers, including one pair of male and female speakers for each of the non-English languages (Finnish, German, and Mandarin) and a couple of male and female English speakers. In particular, the training dataset contains 1 hour of speech in total, with each speaker having 70 utterances, which in total account for less than 7 minutes on average. More details of this dataset are provided in Table 2.1. The use of the VC Challenge training dataset is essential but not enough to train a model such as FastVC. In this project, the VCTK and VCC datasets were simply merged, which is a common and allowed approach in the Challenge.

The evaluation partition of the VC Challenge dataset is disjoint from the training dataset and was released near the end of the Challenge. The Challenge dataset contained 25 new sentences uttered by the source speakers and did not contain transcriptions. The use of manual annotations was allowed for the training partition of the dataset, but not for the evaluation partition. The data for the VC Challenge is sampled at 24 kHz and stored in a 16-bit format.

The need for speaker-independent latents poses a representation learning problem. Ideally, the latent representations found by the encoder network in FastVC disentangle the speaker's information and contain linguistical information. van den Oord et al. (2017) show that the discrete latent representations found by VQ-VAE are similar to phonemes by learning a mapping from the codebook indices of the embeddings used at a given time to the phoneme uttered at the corresponding step. In order to make this comparison, phonetic annotations are needed. Not the VCTK corpus nor the VC Challenge dataset include phonetic annotations, so an additional dataset was needed.

The TIMIT dataset by Garofolo et al. (1993) is chosen for the representation task. This dataset is public and contains speech data and hand-verified time-aligned phoneme transcriptions. In particular, only the test partition of this dataset is used to avoid incorporating a lot of new speakers. This dataset was used for data analysis purposes and is not included in the models trained for the VC Challenge. The TIMIT training partition contains 168 speakers and a total of 1.4h of speech, accounting for an average of 30s per speaker. This data has a sampling rate of 16 kHz and a resolution of 16 bits.

All the experiments use a batch size of 16 and train on 90% of the available data. The remaining 10% is left for testing purposes. Note that since VC is not performed during training, the pair of source and target speakers will be different in inference. However, the data partition is performed for rigorousness, especially for the reported reconstruction results.

The data used in both models come from the previous datasets, which have different sampling frequencies. In both cases, the input waveforms are resampled to match the sampling rate of the model. That is, the inputs to FastVC are resampled to 22050 Hz and the inputs to PhonetVC to 16 kHz. For these two models, the input and output audio is quantized with 16 bits. Note that for both models, one-hot encoded embeddings are used. The use of one-hot embeddings means that the models have different dimensions for each distinct training set and thus need different weights.

3.4.2 System specifications

The FastVC and PhonetVC models have been developed using PyTorch (Paszke et al., 2019), and the code has been tested for Python 3.6,3.7, and 3.8. The I/O operations have been performed using the external TorchAudio library from PyTorch. For additional signal process functionalities, SciPy (Virtanen et al., 2020) and Librosa (McFee et al., 2020) have been used, and NumPy (Oliphant, 2006) has been used for general numerical computation. Additionally, the Phonet toolkit used in PhonetVC uses Keras (Chollet et al., 2015).

All the training was performed on a server with a single Nvidia GeForce GTX 108 GPU with CUDA v9.1.8 and an Intel(R) Core(TM) i7-8700K @ 3.70GHz CPU.

4 Results

This chapter presents the results obtained with the previous models. FastVC is given particular emphasis, for it is the most successful model amongst the tested ones. In particular, FastVC was the chosen model to generate conversions to the VC Challenge.

In particular, the variant of FastVC submitted to the Challenge only uses AE training and has 32-dimensional latents that run at 2.5 Hz. This specific model has around 32 million parameters and a memory footprint of 2.02 GB in conversion mode, and it was the best-rated proposal in a small subjective evaluation test ran at Logitech[1]. For this subjective test, four conversions for the monolingual task and 4 for the cross-lingual task were created for several models. The tested models included AutoVC, the chosen FastVC and one variant with 64-dimensional latents, and the the two VC Challenge baselines: ASR+TTS (see Section 2.2.1) and CycleVAE (see Section 2.1.2). AutoVC was included to compare the success of the modifications of FastVC concerning this model. The Challenge baselines were added to compare our system to the previous state-of-the-art.

FastVC took AutoVC by Qian et al. (2019) as a starting point. One of the main requirements was to speed-up this model while preserving high speech quality. After all the modifications described in Section 3.2 and optimizing the performance of the baseline implementation, the conversions are significantly faster to those obtained with AutoVC. In particular, the conversions are computed around 500x faster than AutoVC and takes 4x less than the duration of the generated audio on CPU[2]. However, this latter has one caveat and is that the whole speech waveform is needed to perform VC with FastVC. Therefore, the speed of results does not imply that real-time continuous VC is possible with the proposed model.

This chapter is structured as follows. First of all, the over-smoothing problem from which AEs, and in particular also FastVC, usually suffer, is illustrated and discussed. Secondly, the pitch contours of some of the samples generated with FastVC are analyzed and compared to those obtained using AutoVC. Then, the latent representations obtained by the AE in FastVC are analyzed. In particular, the analysis confirms that the unsupervised representations are speaker-independent and similar to the human phoneme alphabet as in van den Oord et al. (2017).

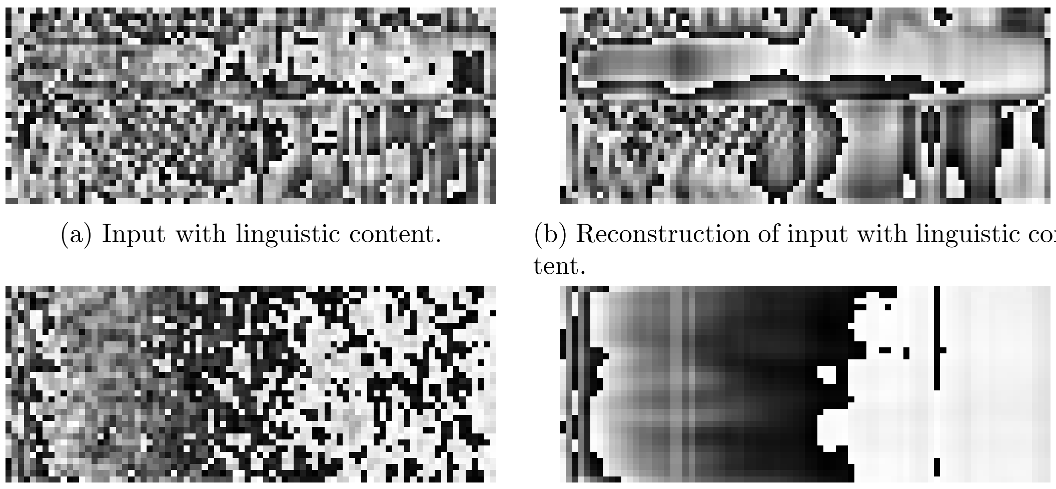
The last part of this chapter evaluates the proposed models. First of all, an objective evaluation framework for VC systems using Perceptual Evaluation of Speech Quality (PESQ), an industry standard for evaluating voice quality, is proposed. Lastly, the subjective evaluations of the VC Challenge models are presented.

4.1 Over-smoothing problem

One common problem of AEs and especially VAEs is that the reconstructions are usually over-smoothed (Chou et al., 2018; Liu et al., 2018; Qian et al., 2019). FastVC and PhonetVC

[1]The subjective test was run in mid-May, and the Challenge submission deadline was at the end of the same month.

[2]Intel(R) Core(TM) i7-8700K @ 3.70GHz CPU.



(a) Input with linguistic content.

(b) Reconstruction of input with linguistic content.

(c) Input without linguistic content.

(d) Reconstruction of input without linguistic content.

Figure 4.1: Comparison between the input and reconstructed mel-spectrograms in the basic configuration of FastVC where only the AE is trained. Examples from segments with and without linguistic information are provided.

follow the approach proposed by Qian et al. (2019) to tackle this problem, which consists in incorporating the PostNet module depicted in Figure 3.3. This module is based on residual networks (He et al., 2015) and is aimed at building the finer details on top of the smoothed output mel-spectrogram. However, the inputs fed to the mel-spectrogram inverted still suffer from over-smoothing.

Inverting over-smoothed mel-spectrogram gives poor quality and buzzy-sounding speech, as noted in Kameoka et al. (2018). Using a generative model for the mel-spectrogram inversion can compensate for such over-smoothing. However, in some cases, learning to compensate for overly smooth conditioning signals requires the fine-tuning of the speech generation models, which sometimes is costly. In particular, MelGAN provides high-quality speech even if used with the pre-trained weights, which have not been trained on the same dataset as FastVC and are not enforced to compensate for the input irregularities. Figure 4.1 shows the over-smoothed outputs of the FastVC model submitted to the VC Challenge, which as discussed in Section 4.5 still provide high-quality speech.

4.2 Pitch inconsistencies

AutoVC achieved state-of-the-art results by disentangling of the speaker's identity in the latent representations. However, Qian et al. (2020) shows that prosodic information leaks through the bottleneck, causing the target pitch to fluctuate unnaturally. To tackle this issue, Qian et al. (2020) proposed to remove not only the speaker identity from the latent representations but also the prosodic information.

The approach to achieve latents that do not contain prosody information is the same as the one followed in Qian et al. (2019) to achieve speaker independence. Qian et al. (2020) propose to feed uncompressed pitch information to the decoder, so the latent representations do not contain this information according to the redundancy principle Barlow (1989). The disentangling of this information is again sensitive to the hyperparameters governing the information bottleneck. In particular, the prosody information fed to the decoder corresponds to a pitch estimation computed for each frame.

In conversion, the prosody of the source should be taken into account, for it being paralinguistical information that can change the meaning of the utterance. To match the target speakers' prosody style, Qian et al. (2020) propose applying the simple transformation described in (2.5). Using this transformation effectively maintains the source waveform intonation but provides the target speaker's characteristics.

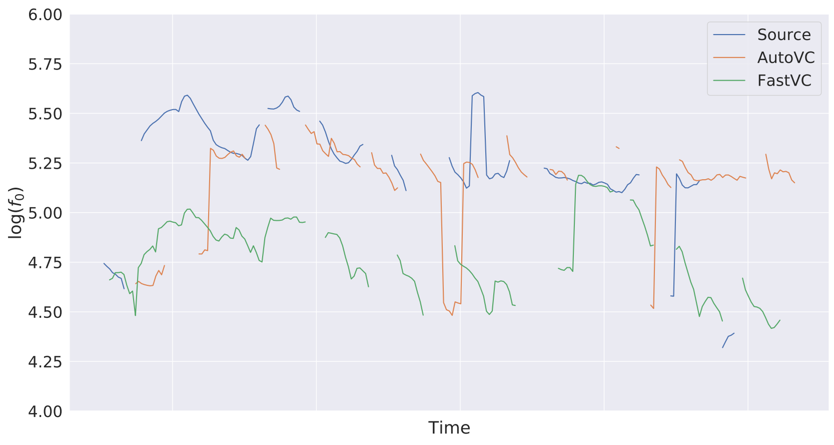
The temporal downsampling factor proposed in Qian et al. (2020) is larger than that used for the original AutoVC, given that the latent representations in the former model contain more information than in the latter. The downsampling factor matches the design choice of the FastVC model submitted to the VC Challenge, which does not suffer from the unnatural fluctuation of the pitch. This result suggests that the latent representation rate proposed in Qian et al. (2019) was too high, thus allowing the filtering of speaker information when combined with the proposed dimensionality reduction bottleneck.

4.3 Exploring the latent representations

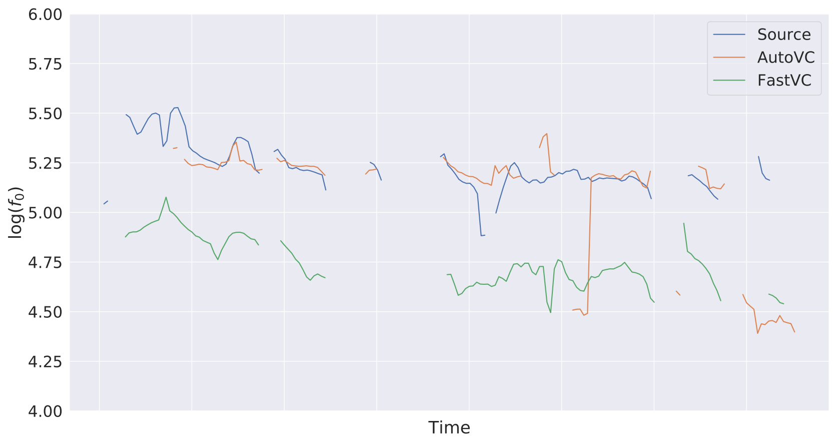
An interesting topic that arises from FastVC is the representation learning problem based on extracting speaker-independent features in an unsupervised fashion. van den Oord et al. (2017) claimed that the discrete latents obtained with VQ-VAE were closely related to phoneme. In particular, a 49.3% accuracy in phone classification based on the latents was achieved, while prior most likely phoneme gives 7.2% accuracy. A more complete analysis of the VQ-VAE representations is performed in Chorowski et al. (2019). In this paper, the unsupervised extraction of meaningful latent representations from the speech is studied. The aim is to learn high-level representations that can capture the linguistic content of the signal while being robust to unwanted low-level variations.

As described in Section 3.4.1, the TIMIT (Garofolo et al., 1993) dataset is used to identify the linguistic content in the latent representations. Similarly to van den Oord et al. (2017); Chorowski et al. (2019), the approach is to predict the phoneme from the latent representations. A simple perceptron is used to find the existence of a hypothetically simple correspondence between phonemes and latents. This model is trained using the latent representations extracted from the TIMIT used data with a trained FastVC network. The obtained latents are split into train, validation, and test partitions accounting for the 70%, 10%, and 20%, respectively.

The information bottleneck on the temporal dimension chosen for the FastVC model used in the Challenge yields a latent representation with a 2.5 Hz rate. This rate is a factor of 10 lower than the rate of the latents in VQ-VAE. The average phoneme rate is around 10 Hz Van Kuyk et al. (2017); Cernak et al. (2017), which means that each latent vector at a given time represents more than one phoneme. Each latent vector had to be assigned a label to train a classifier on top, which poses a problem since, on average, each latent comprises around four phonemes. Three approaches were followed, involving the classification of latent vectors into phonemes, diphones, and triphones. Phonemes are a unit of sound that allows distinguishing words in a language. Diphones are an adjacent pair of phones, and triphones represent three consecutive phonemes. When considering phonemes, each latent was assumed to represent the phoneme with a larger intersection in the temporal domain. The same approach was followed



(a) Pitch contour with source speaker SEF2 (English female), content E10004 and target speaker TEM2 (English male).



(b) Pitch contour with source speaker SEF1 (English female), content E10004 and target speaker TGM1 (German male).

Figure 4.2: Comparison of pitch contours of source speech signal and conversions with AutoVC (Qian et al., 2019) and FastVC, our model. The source and target speakers used for the conversions in these plots belong to the VCC20 dataset. The pitch contour has been computed using the algorithm implemented in Jadoul et al. (2018).

Classifier	**Accuracy**
Linear classifier on raw latents	42.45%
Linear classifier on quantized latents (256 clusters)	40.09%
Linear classifier on quantized latents (128 clusters)	34.43%
Linear classifier on quantized latents (41 clusters)	31.13%
Random classifier	2.44%
Prior most likely phoneme on train partition	9.43%

Table 4.1: Results of the mapping from phonemes to latents. The latents were computed using FastVC on the test partition suggested in the TIMIT dataset (Garofolo et al., 1993). The resulting latents were then randomly split into train (70%), validation (10%) and test partitions (20%). The perceptron was trained with cross-entropy loss using SGD with learning rate 0.01 and momentum 0.9. The training stopped the first iteration that the loss increased in the validation partition. VQ was performed using the implementation of Pedregosa et al. (2011) for 41 (the number of distinct phonemes), 128 (the value used in VQ-VAE) and 256 clusters.

using diphones and triphones. Contrarily to VQ-VAE, the latent representations obtained with FastVC are not quantized. The classification accuracy of the latents quantized using VQ is also reported to compare how this quantization affects. This quantization is performed offline and not incorporated in the training of the latents as in van den Oord et al. (2017).

The phoneme classifier was trained by minimizing the cross-entropy loss with Stochastic Gradient Descent (SGD) and early stopping on the loss on the validation partition. The classification accuracies on phonemes computed over the test partition are reported in Table 4.1. Regarding the prediction of diphones, the classification accuracy of the perceptron was 33.33%, while a random classification has an accuracy of 0.45%. The classification accuracy of a classifier based on always choosing the most probable class is 17.78%. The results obtained with triphones were not significantly better than those obtained with the most probable class classifier. Even if the latent representations' low rate suggested that a latent represents a combination of sounds rather than a single phoneme, the number of distinct units with groups of phonemes exponentially grows with the group size. This growth implies that there are more classes to predict, and instances of some of them may not even be seen during training. The latter justifies the performance drop when predicting diphones and triphones instead of single phonemes.

The results in Table 4.1 suggest that there is indeed a correspondence between latents and phonemes. Such correspondence is preserved when the latent representations are quantized with 256 clusters. However, for quantizations with less resolution, the performance drop is significant. For comparison, VQ-VAE van den Oord et al. (2017) uses a 128-dimensional discrete space and obtains a classification accuracy of 49.3%, while a choosing the prior most likely phoneme gives a 7.2%. A classification drop from the results in van den Oord et al. (2017) is expected due to the lower rate representation. The details of the phoneme classifier and its training are not shared in van den Oord et al. (2017), which in case of not being a perceptron, could further justify the performance drop.

Overall, the results of the phoneme mapping are surprising given the hard assumption on the label assignment and the very low rate of the latent representation. These results suggest that the latent representations are similar to the human phonetic alphabet and, hence, are strictly related to the linguistic information as hypothesized.

FastVC was built on the hypothesis that the latent representations are speaker-independent.

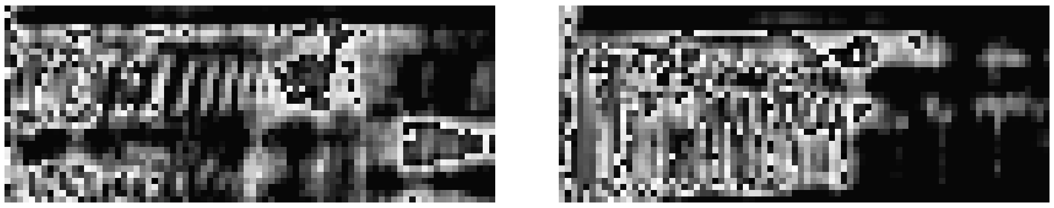


Figure 4.3: Learned transformation for segment with linguistical content.

Speaker independence is achieved by the redundancy principle, as mentioned in Section 3.2. One of the variants of FastVC consisted of building a speaker classifier to enforce such independence implicitly (see Section 3.2.2). This speaker classifier is trained simultaneously as FastVC and provides a learning signal intended to remove the speaker's information from the latent representation. To test whether this classifier is useful, the speaker predicted from the latent representation is reported. In this case, the hypothesis is not that there is a simple correspondence as it was the case for the last experiment, but only the prediction capabilities are tested. Therefore, the speaker classifier trained along with FastVC was used instead of a new simpler model. In particular, the speaker classifier consists of a 3-layer CNN with averaging over the time dimension (Mor et al., 2019).

The speakers were predicted using this classifier with the latents obtained with a trained FastVC. Even if the classifier network was trained at the same time as FastVC, the prediction accuracy was 0% when the speaker-independence signal was used on the latents and when it was not, which confirmed our hypothesis. These results were obtained with a model that was trained using 278 speakers. These speakers belong to the VCTK corpus (Veaux et al., 2016) and the test partition of the TIMIT dataset (Garofolo et al., 1993). The speaker-independence results suggest that the redundancy principle is enough to achieve speaker-independence, which is in line with the results reported in Qian et al. (2019, 2020).

FastVC allows the training with raw waveforms as input. The techniques to train this module are proposed in Section 3.1, and are aimed at learning problem-dependent transformations better suited than the widely used general-purpose mel-spectrogram. In particular, the representations found with the transformation module of FastVC are sparser than the mel-spectrogram in general. Moreover, the transformation seems to capture the difference between speech with and without linguistic content as shown in Figure 4.3 and Figure 4.4.

Note that the transformation for segments with linguistic content is slightly sparser but very similar to a mel-spectrogram (see Figure 4.1 for reference). However, the noise segments that lack linguistic content have very few non-zero coefficients (zeroes represented with black). These results are consistent with the hypothesis that the latent representation found by the encoder of FastVC encodes the linguistic content. Overall, learning the transformation from raw waveforms can allow a better allocation of the space resources to the essential features for a given task. Moreover, the learned distinction between segments with and without linguistic content can act as an in-built Voice Activity Detector (VAD) that can likely reduce the output noise during silences.

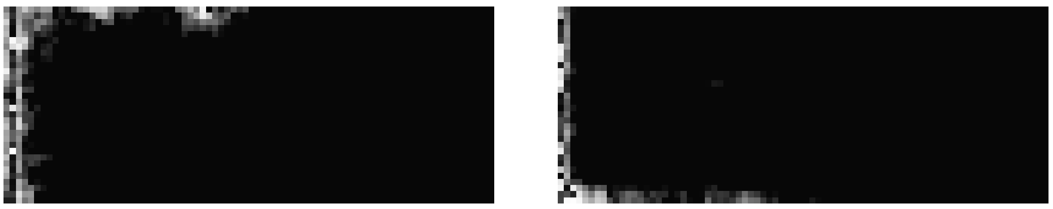


Figure 4.4: Learned transformation for noise segment.

4.4 Objective evaluation

One of the main difficulties in building style transfer models is that there are no standardized objective measures (see Chapter 1). The lack of such metrics hinders the system comparison and the performance of ablation studies. In this section, some objective measures are proposed to evaluate the proposed models. In particular, the proposed measures use PESQ, an objective method that rates the speech quality by predicting the Mean Opinion Score (MOS).

The MOS is a quality measure ranging from 1 to 5, where 1 and 5 represent bad and excellent quality, respectively. This score is subjective and based on the average rating given by human evaluators. In particular, this subjective metric is widely used when testing speech generation systems and is the metric used for the model comparison in the VC Challenge.

As discussed in Chapter 2, using parallel datasets simplifies the VC problem and in general yields better results than non-parallel VC. The reason is that parallel datasets provide target utterances for the converted samples. Even if such samples are not the unique solution, they are enough to learn a conversion function that generalizes to unseen utterances.

The fact that PESQ is not used as a standardized measure to substitute MOS is that the former requires both the desired waveform and the one generated with the evaluated system. The evaluation in PESQ is performed sample-by-sample, and thus is adequate for evaluating TTS systems or speech coding techniques but not suited for the evaluation of VC systems at all.

Even if there is no known desired waveform in the VC problem, this metric is performed on 100 parallel utterances from the testing partition of FastVC. As mentioned before, when training with parallel data, the desired waveform is not unique, so low PESQ results are expected. To match the length of the converted and the target waveforms, DTW is applied. In particular, FastDTW from Salvador and Chan (2007) is used.

The results with this objective evaluation are shown in Figure 4.5. The PESQ values are very similar for all the different systems and have a large variance. The best system, according to this comparison, is the variant of FastVC with cyclic training. However, this system yields poor conversions, as mentioned in Section 3.2.2. This suggests that the PESQ evaluation is not suited for the VC model comparison. This can be justified by the ill-posedness of the problem; a valid output other than the parallel utterance may be obtained. Therefore, high-level descriptors would be needed for this case to account for valid outputs that differ from the parallel utterance at signal level.

The approach that FastVC takes to deal with non-parallel data is to learn the conversion function on the task of speech reconstruction. The reconstruction performance alone is not a useful metric to evaluate a VC system because it does not measure the speaker's disentanglement. In particular, perfect reconstruction can be achieved if there is no information bottleneck at all

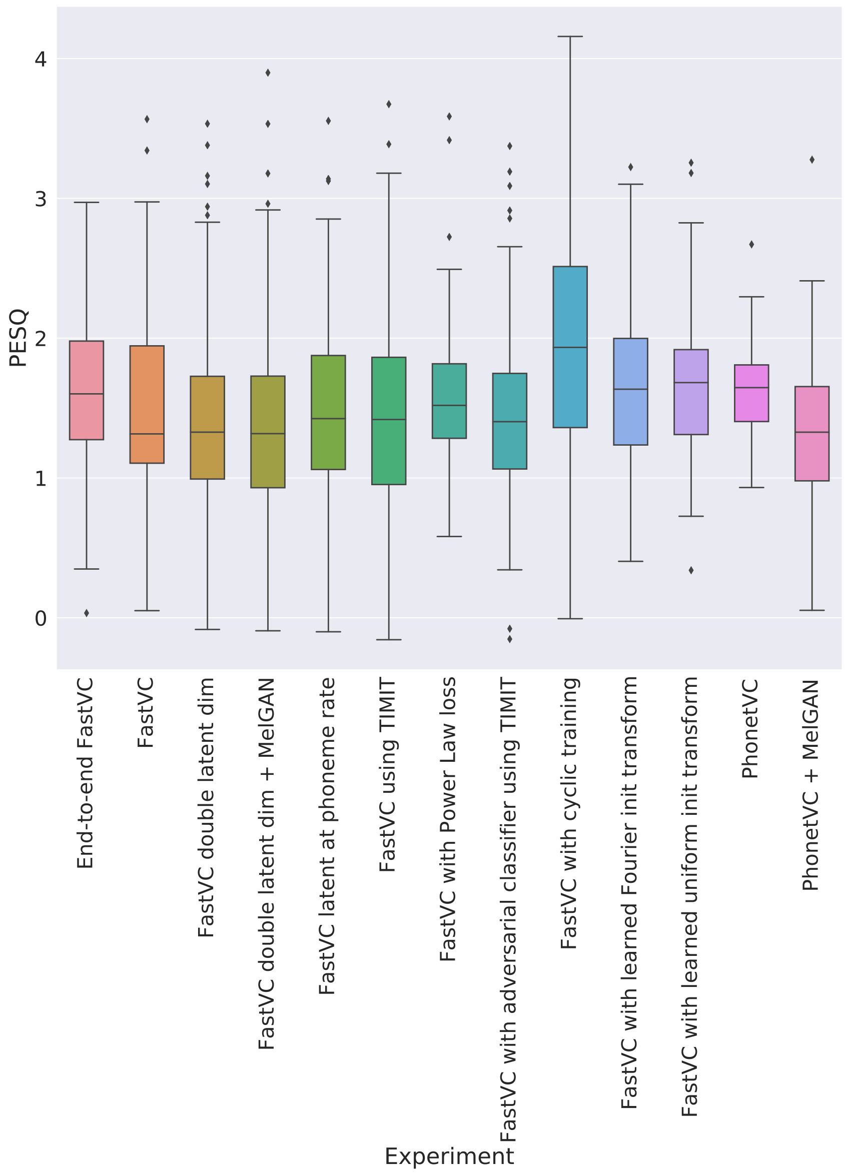


Figure 4.5: Boxplot of parallel conversion PESQ.

and thus also no speaker-independent latents.

In this case, the PESQ measure is more suited since self-reconstruction was learned during training, and mapping to the same speaker is a valid VC instance. The results are presented in Figure 4.6. Note that this comparison rates FastVC as the best model, which is consistent with the subjective evaluation performed in the company.

The results on reconstruction also show that, as mentioned in Section 3.2.2, the variant of FastVC with cyclic training does not yield better results. With the proposed evaluation, this modification shows an enormous variance, with that model giving the best and the worst individual results.

FastVC with end-to-end training performs worse in terms of PESQ than FastVC. This can be justified because in the end-to-end training, the aim is not to match the input mel-spectrogram but to maximize the GAN objective. A future subjective evaluation would be needed to confirm

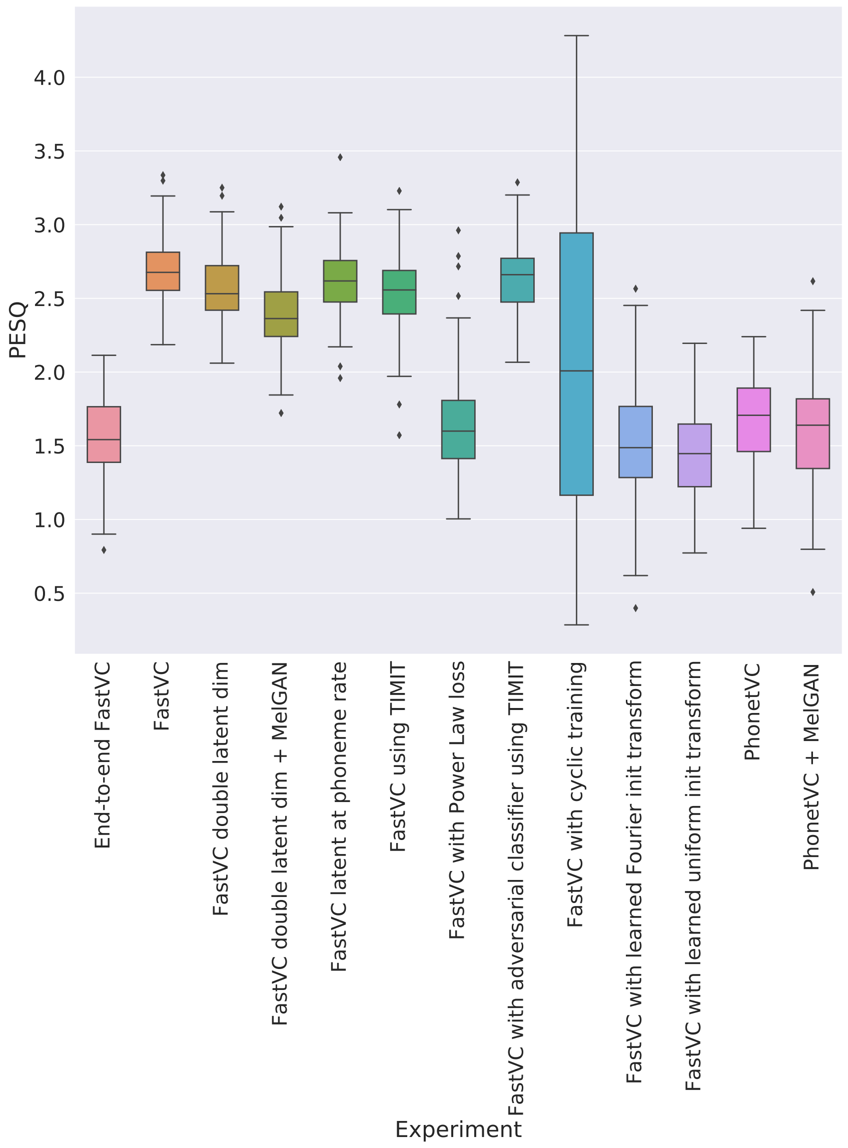


Figure 4.6: Boxplot of reconstruction PESQ.

if the PESQ also correlates with the perceived quality in such cases.

4.5 VC Challenge results

The problem with comparing the proposed models to the state-of-the-art is solved with the large-scale crowd-sourced perceptual evaluations performed in the VC Challenge. According to the feedback received on the subjective test ran at Logitech, the best model for the monolingual task was ASR+TTS (see Section 2.2.1). The success of this solution in the monolingual task is not surprising given the high-quality ASR and TTS English systems that are already found at the consumer level. Since FastVC was perceptually not better to the Challenge baselines for the monolingual task, the participation in this task was withdrawn.

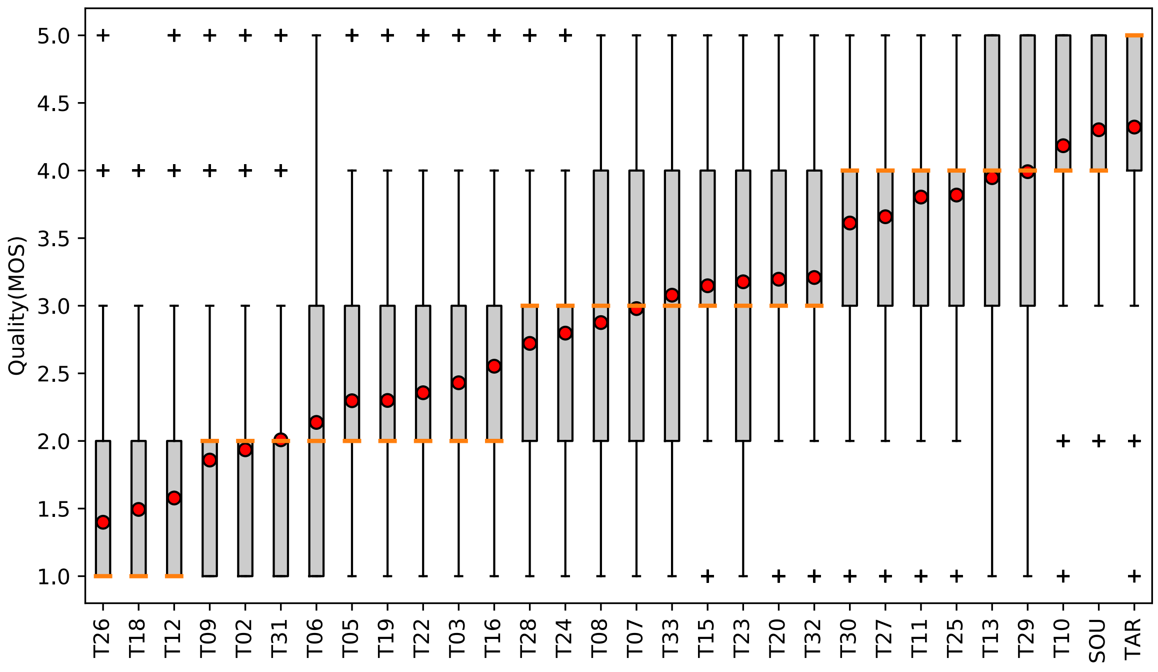
Regarding the cross-lingual task, FastVC reported the best results. Therefore, the converted samples generated for this task were submitted to the Challenge. From more than 90 participant teams, only 33 teams submitted samples, including the two Challenge baselines and the VC Challenge 2018 winner. Out of these teams, 11 come from the industry, as is the case of FastVC. The other teams divide into 17, coming from academic institutes, and two from personal participation. Thirty-one teams participated in the monolingual conversion task, whereas 28 teams participated in the cross-lingual conversion task.

The subjective evaluations are performed using MOS. The evaluations test the naturalness, which is the preservation of the source content, and the similarity to the target speaker. These evaluations are performed by 206 Japanese listeners and 124 English listeners.

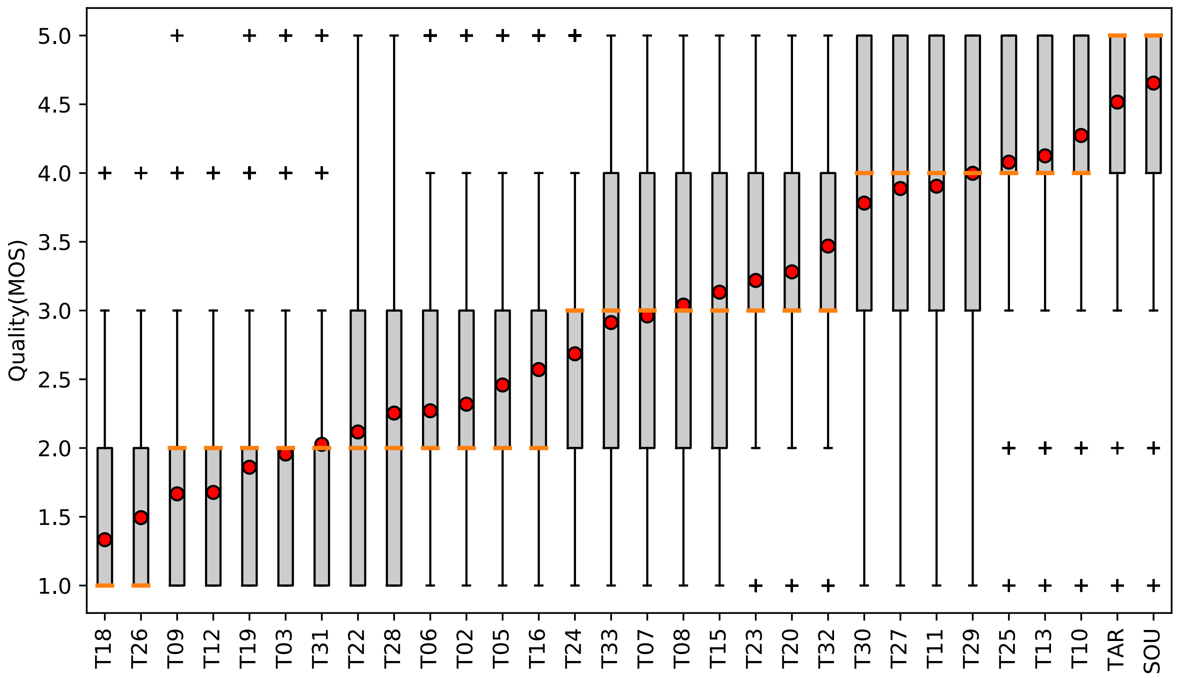
The results in the Challenge are anonymized. The label **T15** represents FastVC. The Challenge baselines, CycleVAE and ASR+TTS, are the teams **T16** and **T22**. The identity of the winner of the VC Challenge 2018 was also revealed, and corresponds to **T11**.

Figure 4.7 depicts the naturalness results classified by the evaluators' language. In both cases, FastVC outperformed the Challenge baselines but had worse results than the past Challenge laureate, which scored fifth in this year's edition.

Figure 4.8 shows the similarity results, were FastVC downgraded concerning the naturalness results. The poor matching of the target speaker characteristics can be justified because the dataset is imbalanced. As discussed in Section 3.4.1, the VC Challenge dataset contains few data, which is not enough to train FastVC. The former dataset was combined with the VCTK corpus by Veaux et al. (2016), which contains around 3.5x more data per speaker. This problem is further emphasized, considering that most of this data comes from English speakers. In this case, there are around 45h of English speech and only 14 min of the non-English languages included in the cross-lingual task. Potential solutions to this problem are discussed in Chapter 5.

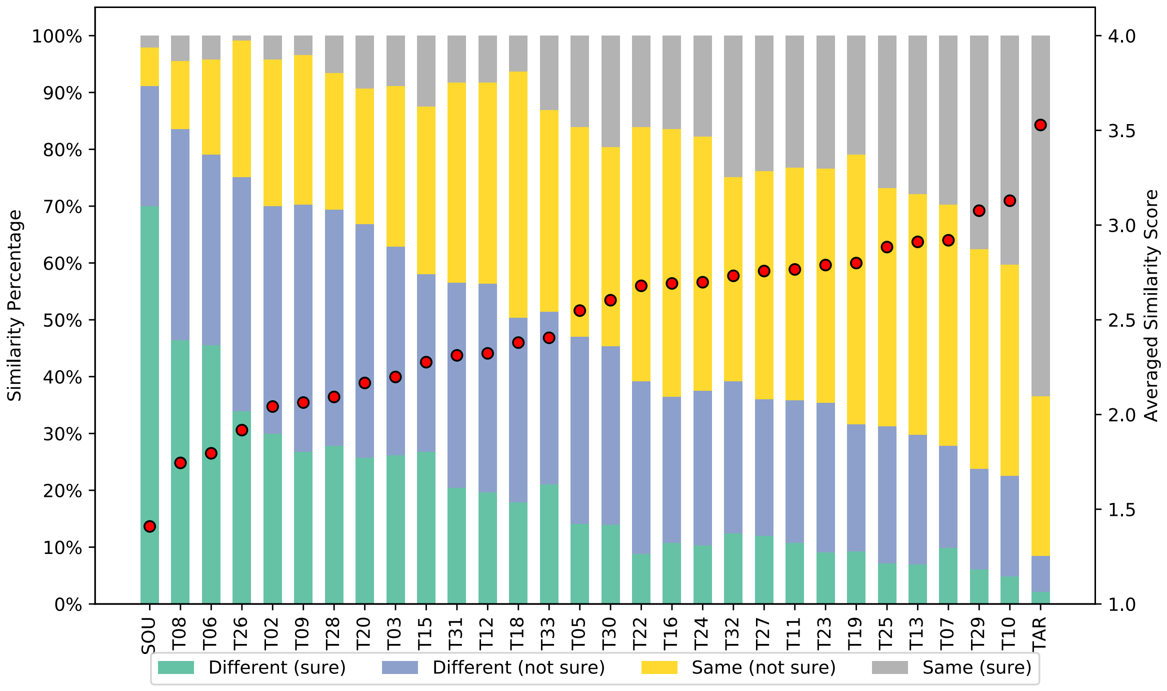


(a) Japanese listeners.

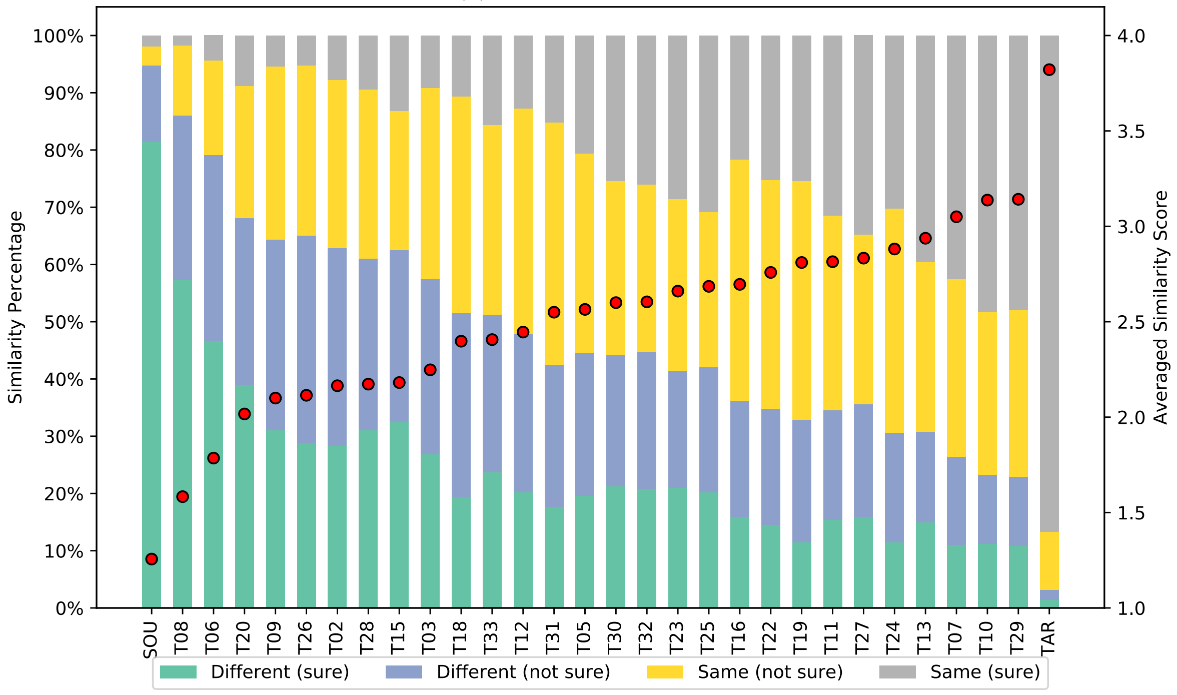


(b) English listeners.

Figure 4.7: Naturalness results for the cross-lingual task of the Voice Conversion Challenge 2020. MOS scores are arranged in accordance with their mean (red dot).

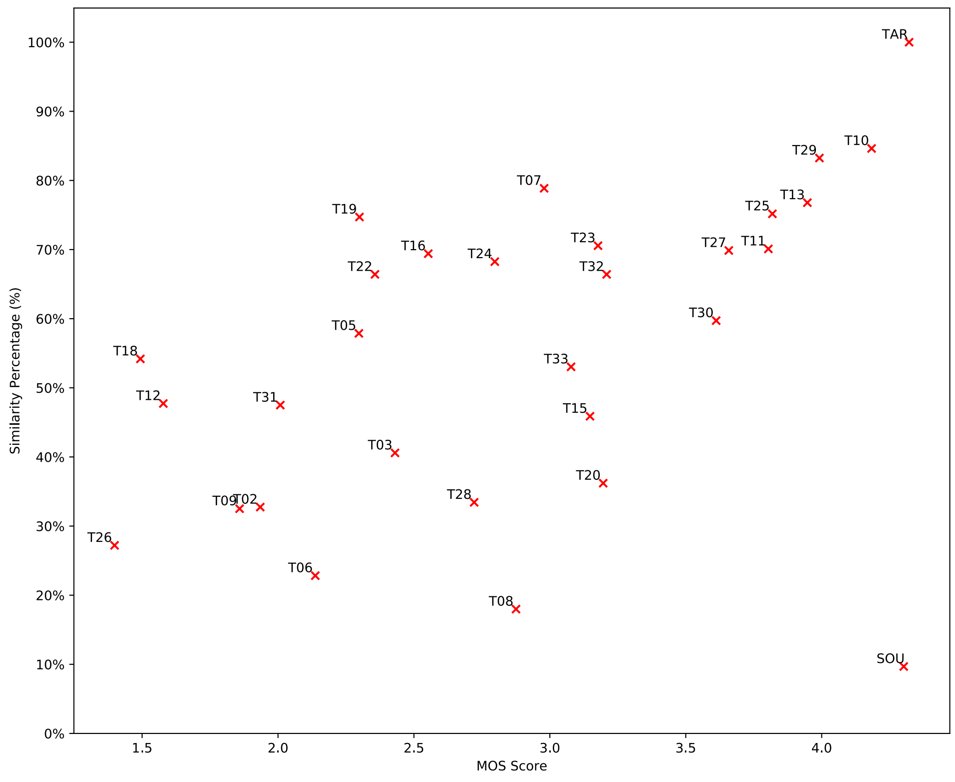


(a) Japanese listeners.

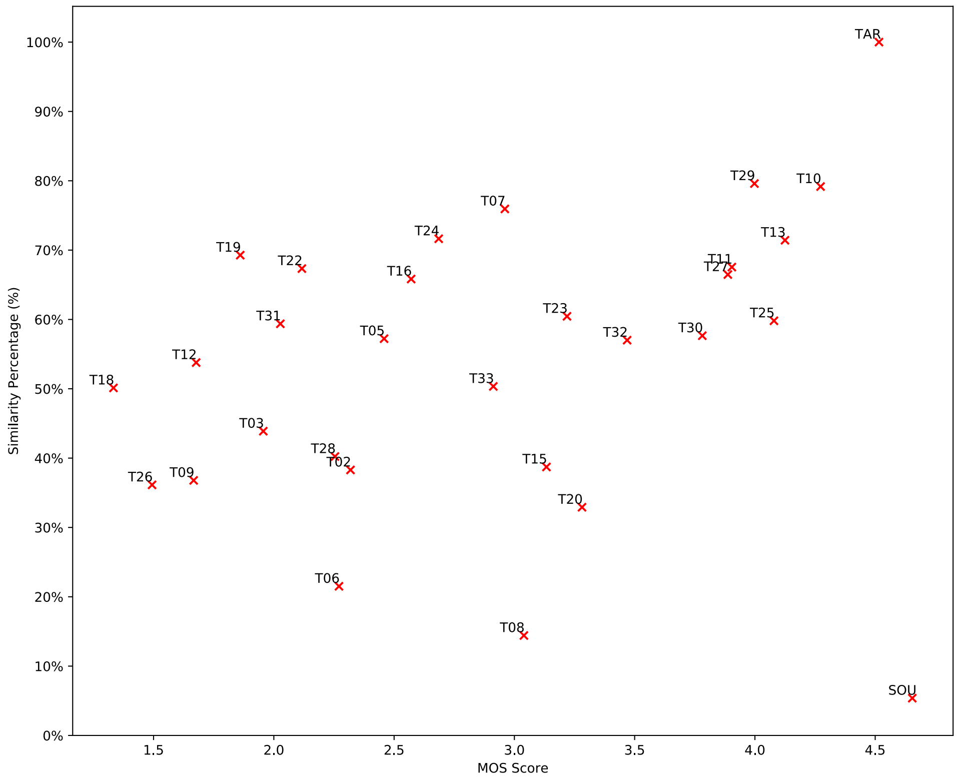


(b) English listeners.

Figure 4.8: Similarity results for the cross-lingual task of the Voice Conversion Challenge 2020. Similarity scores are arranged in accordance with their mean (red dot).



(a) Japanese listeners.



(b) English listeners.

Figure 4.9: Scatter plot matching naturalness and similarity scores to the target speaker for the cross-lingual VC task when averaging all speaker pairs.

5 Conclusions and future work

This work proposed FastVC, an end-to-end non-parallel system for cross-lingual VC. FastVC has a simple AE structure trained on speech reconstruction but outperforms the VC Challenge baselines on the cross-lingual task of this year's Challenge in terms of naturalness. FastVC can convert audios of arbitrary lengths and perform many-to-many VC with a single network.

FastVC only trains on speech waveforms and speaker identities, so it requires no annotations. Instead, some of the models submitted to the Challenge required transcriptions, linguistic information, or supra-segmental information such as the intonation. The use of these features either requires manually annotated datasets, which are scarce, or automatic labeling systems. Some of the Challenge participants reported using manual labeling and verification of such transcriptions for its later use.

A few approaches amongst the systems that submitted to the Challenge also involved temporally aligning the transformed features using methods such as DTW and using pitch-synchronous algorithms. Other systems used excitation and phase prediction and correction techniques to improve the waveform synthesis. Additionally, some approaches apply a variety of pre-processing and post-processing techniques.

Compared to other approaches, FastVC is a straightforward model that works on unprocessed raw data and has no external annotations. Instead, all the representations are computed from data in an unsupervised fashion with a simple AE structure. FastVC is also language-independent, which means that its behavior does not change with different input languages. According to the subjective evaluation performed in the company, the ASR+TTS baseline attained the best performance on the monolingual VC task. However, the perceived performance of this approach substantially decreased for the cross-lingual task. The performance drop is justified with the language dependence of the model, which relies on intermediate text annotations.

A given text may be uttered in different ways for distinct languages, so text annotations cannot be used in the cross-lingual task. The text annotations may be mapped to phonemes, but some phonemes also vary from one language to another. In the case of Mandarin, an additional difficulty arises for this model. Mandarin characters are ideographic, so a previous conversion to Pinyin, the romanization system for Mandarin, is needed.

Compared to the former approach, FastVC preserves timing information, which is not stored in the text transcription but is a para-linguistical feature that may change the meaning of an utterance. FastVC also preserves prosodic features but could be disentangled from them using the same redundancy principle used to achieve speaker-independent latents. This approach's success is discussed in Qian et al. (2020), which opens a new window to the general style transfer problem. Following these ideas, information bottlenecks on the input signal and uncompressed style information can be used to obtain style-free latent representations for different definitions of style and content.

This project also explored the use of speaker-independent features. Similarly to the ASR+TTS, PhonetVC first extracts speaker-independent features related to the uttered sound. To do so, it relies on Phonet by Vásquez-Correa et al. (2019). These features are later merged with the target speaker information to perform VC. Another approach would be to use other speaker-independent representations such as the AM part of an AM-FM decomposition. This representation is explored in Motlicek et al. (2020). In particular, the AM decomposition is shown to be related to content and FM decomposition to the speaker. This decomposition could potentially be used to avoid finding speaker-independent latent representations and speaker representations.

FastVC is presented with several variants that take inspiration from the most successful VC systems. This work proposes the use of objective scores to compare them, which differs from the general trend only to report subjective scores in VC. In particular, the PESQ score on the reconstructed speech is a good indicator of the performance when applied to AE frameworks with speaker-independent latents. This score does not require the participation of human evaluators and is not subjective and potentially biased depending on the population of evaluators.

This project also explores the unsupervised latent representations extracted with the proposed model, which disentangle the speaker's information in an unsupervised way and are closely related to phonemes. The speaker independence test could potentially be used with the PESQ objective measure for an exhaustive evaluation of VC models.

Despite the good results in naturalness exhibited by FastVC in the VC Challenge, this model is worse than the state-of-the-art at capturing the speaker's style of speakers with few data. As discussed in Section 4.5, this is justified with the fact that the training dataset is very imbalanced concerning the language, and the performance could be degraded for non-English speakers. One possible approach to tackle the language imbalance problem is to incorporate additional non-English speech datasets to balance the languages.

The approach of adding additional datasets for the minority languages adds even more speakers, which reduces the problem to data imbalance only concerning the data per speaker. However, the percentage of data per speaker on the VC Challenge would be even smaller in this case. A different approach to tackle dataset imbalance is to compute the loss only over the Challenge data and use the performance on the larger dataset as a regularizer to enforce generalization. This approach is referred to as the multi-reader technique and described in Wan et al. (2017).

The use of speaker embeddings computed from waveforms instead of using one-hot representations could also potentially solve the problem of having few data per speaker. Following the interpretation in (Barbany, 2018, Section 3.3.), such representations decompose the speaker's characteristics into a basis of speakers. Therefore, having speakers with similar characteristics in this space could help the model achieve good performance with few data per speaker. However, the implementation of this approach on AutoVC was reported to give worse results in the subjective evaluation. The performance was reported to further drop with languages not seen during training. Additionally, achieving good speaker representation needs a wide variety of speakers, (Jia et al., 2018).

One of the main objectives of this work was to implement fast high-quality VC systems. Given the speed of FastVC, the company showed interest in achieving real-time continuous VC. FastVC uses the whole audio waveform as input during the conversion phase, but real-time systems require working with audio chunks. Two approaches were followed for the real-time generation, including taking non-overlapping chunks and overlapping windows. After the conversion of the overlapping windows, these were merged using the overlap-add method. However, the VC results obtained with the window values required for real-time speech applications failed at capturing the target speaker's style. These adverse results are justified with the very low rate of the latents, which could be solved using lower downsampling factors. This higher rate should be compensated with more dimensionality reduction or the incorporation of additional information bottlenecks such as VQ.

As mentioned in Section 3.2, choosing the correct information bottleneck is crucial. However, there are no theoretical results for such values. The problem of non-parallel VC was reduced to a speech reconstruction problem in FastVC. To set the rate and dimension of the latent representations, these could be interpreted as the features extracted in a speech coding scheme. Wang and Kuo (1998) proposed a low rate speech coding scheme using classical methods that operates at 800bps, which is the lowest bitrate of classical speech codecs. With this result in mind, one could determine the theoretical information bottleneck needed only to encode linguistic information.

Choosing the latents' rate to be that of the phonemes, which is around 10 Hz Van Kuyk et al. (2017); Cernak et al. (2017), one could represent each latent using VQ with as many clusters as distinct phonemes there are. A similar analysis of speech signals is detailed in Cernak et al. (2017), where a theoretical information rate of speech is proposed. In particular, this work finds an upper bound of around 100 bps. The gap between this value and the codec proposed in Wang and Kuo (1998) could be filled with speech representations extracted using similar techniques as FastVC.

The latent representations obtained with FastVC ideally preserve the salient features of the encoded data and are invariant to nuisance low-level signal details. This latent signal representation allows disentangling different factors of variation in the data and discarding spurious patterns. Following from this latter, FastVC could be used in denoising. Since the latent representations have a lower rate than the raw speech, they will not store the randomness of the signal but only the critical information that allows for proper reconstruction of the mel-spectrogram in the ℓ_2-norm sense.

The main flaw of the proposed architecture is that FastVC is not explicitly trained for the task of VC due to the lack of parallel utterances. Instead, FAstVC learns a coding scheme that disentangles the linguistic and non-linguistic components of the speech, relying on information bottlenecks and the redundancy principle.

A more natural approach goes back to the original style transfer problem formulation. In the case of VC, we can identify the content as the linguistic information of the speech signal and the speaker as its style (see Chapter 1). One way to test the performance in the VC task is to evaluate both of them using ASR and speaker classifier systems. This approach could use systems that implement a differentiable mapping so that one could train a model with first-order methods.

The idea would be simple: since the proposed architecture is expressive enough, one could use the same model or a variant of it. Instead of training on reconstruction, however, the model could be trained using the gradients provided by the ASR and classifier systems. This training scheme is depicted in Figure 5.1.

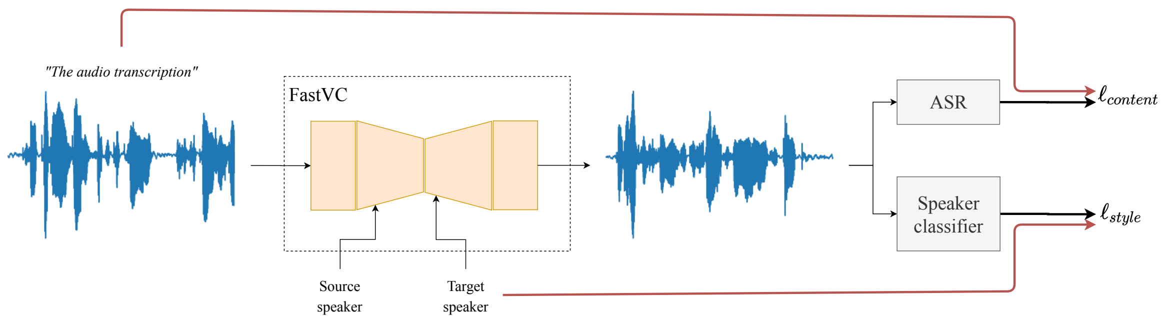


Figure 5.1: Scheme for training FastVC explicitly on VC. The ASR and speaker classifier have frozen parameters; they are only used for the content and style losses. These are computed using the audio transcription and target speaker respectively.

Bibliography

Abe, M., Nakamura, S., Shikano, K., and Kuwabara, H. (1988). Voice conversion through vector quantization. In *ICASSP-88., International Conference on Acoustics, Speech, and Signal Processing*, pages 655–658 vol.1.

Barbany, O. (2018). Multi-Speaker Neural Vocoder. Bachelor's thesis, Universitat Politècnica de Catalunya.

Barlow, H. (1989). Unsupervised learning. *Neural Computation*, 1(3):295–311.

Beckmann, P., Kegler, M., Saltini, H., and Cernak, M. (2019). Speech-VGG: A deep feature extractor for speech processing.

Bengio, Y., Léonard, N., and Courville, A. C. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432.

Butterworth, S. (1930). On the Theory of Filter Amplifiers. In *Experimental Wireless and the Wireless Engineer*, pages 536–541.

Cernak, M., Asaei, A., and Hyafil, A. (2017). Cognitive speech coding examining the impact of cognitive speech processing on speech compression. *IEEE Signal Processing Magazine*, 35(3):97–109.

Chollet, F. et al. (2015). Keras. https://github.com/fchollet/keras.

Chorowski, J., Weiss, R. J., Bengio, S., and van den Oord, A. (2019). Unsupervised speech representation learning using WaveNet autoencoders. *CoRR*, abs/1901.08810.

Chou, J., Yeh, C., Lee, H., and shan Lee, L. (2018). Multi-target Voice Conversion without Parallel Data by Adversarially Learning Disentangled Audio Representations.

Daskalakis, C., Ilyas, A., Syrgkanis, V., and Zeng, H. (2017). Training GANs with Optimism. *CoRR*, abs/1711.00141.

de Chaumont Quitry, F., Tagliasacchi, M., and Roblek, D. (2019). Learning audio representations via phase prediction.

Dieleman, S. (2020). Generating music in the waveform domain. https://benanne.github.io/2020/03/24/audio-generation.html.

Doi, H., Toda, T., Nakamura, K., Saruwatari, H., and Shikano, K. (2014). Alaryngeal Speech Enhancement Based on One-to-Many Eigenvoice Conversion. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):172–183.

Dumoulin, V., Shlens, J., and Kudlur, M. (2016). A learned representation for artistic style. *CoRR*, abs/1610.07629.

Engel, J., Hantrakul, L. H., Gu, C., and Roberts, A. (2020). DDSP: Differentiable Digital Signal Processing. In *International Conference on Learning Representations*.

Engstrom, L. (2016). Fast style transfer. https://github.com/lengstrom/fast-style-transfer/. commit 64.

Felps, D., Bortfeld, H., and Gutierrez-Osuna, R. (2009). Foreign accent conversion in computer assisted pronunciation training. *Speech Communication*, 51(10):920,932.

Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G., Pallett, D. S., Dahlgren, N. L., and Zue, V. (1993). TIMIT Acoustic-Phonetic Continuous Speech Corpus LDC93S1. Philadelphia: Linguistic Data Consortium.

Gatys, L. A., Ecker, A. S., and Bethge, M. (2015). A Neural Algorithm of Artistic Style.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Networks.

Griffin, D. and Jae Lim (1984). Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243.

Hayashi, T., Yamamoto, R., Inoue, K., Yoshimura, T., Watanabe, S., Toda, T., Takeda, K., Zhang, Y., and Tan, X. (2019). ESPnet-TTS: Unified, Reproducible, and Integratable Open Source End-to-End Text-to-Speech Toolkit.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

Heck, M., Sakti, S., and Nakamura, S. (2016). Unsupervised Linear Discriminant Analysis for Supporting DPGMM Clustering in the Zero Resource Scenario. *Procedia Computer Science*, 81:73–79.

Inaguma, H., Kiyono, S., Duh, K., Karita, S., Soplin, N. E. Y., Hayashi, T., and Watanabe, S. (2020). ESPnet-ST: All-in-One Speech Translation Toolkit. *arXiv preprint arXiv:2004.10234*.

Ito, K. and Johnson, L. (2017). The lj speech dataset. https://keithito.com/LJ-Speech-Dataset/.

Jadoul, Y., Thompson, B., and de Boer, B. (2018). Introducing Parselmouth: A Python interface to Praat. *Journal of Phonetics*, 71:1–15.

Jia, Y., Zhang, Y., Weiss, R. J., Wang, Q., Shen, J., Ren, F., Chen, Z., Nguyen, P., Pang, R., Moreno, I. L., and Wu, Y. (2018). Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis.

Johnson, J., Alahi, A., and Li, F. (2016). Perceptual losses for real-time style transfer and super-resolution. *CoRR*, abs/1603.08155.

Kadioglu, B., Horgan, M., Liu, X., Pons, J., Darcy, D., and Kumar, V. (2020). An empirical study of Conv-TasNet.

Kameoka, H., Kaneko, T., Tanaka, K., and Hojo, N. (2018). StarGAN-VC: Non-parallel many-to-many voice conversion with star generative adversarial networks.

Kawahara, H. (2006). STRAIGHT, exploitation of the other aspect of VOCODER: Perceptually isomorphic decomposition of speech sounds. *Acoustical Science and Technology*, 27(6):349–353.

Kim, C. and Stern, R. M. (2010). Feature extraction for robust speech recognition based on maximizing the sharpness of the power distribution and on power flooring. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4574–4577.

Kingma, D. P. and Ba, J. (2014). ADAM: A Method for Stochastic Optimization.

Kinnunen, T., Lorenzo-Trueba, J., Yamagishi, J., Toda, T., Saito, D., Villavicencio, F., and Ling, Z. (2018). A Spoofing Benchmark for the 2018 Voice Conversion Challenge: Leveraging from Spoofing Countermeasures for Speech Artifact Assessment.

Kumar, K., Kumar, R., de Boissiere, T., Gestin, L., Teoh, W. Z., Sotelo, J., de Brébisson, A., Bengio, Y., and Courville, A. C. (2019). MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis. In *Advances in Neural Information Processing Systems 32*, pages 14910–14921. Curran Associates, Inc.

Kumar, S. A. and Kumar, C. S. (2016). Improving the intelligibility of dysarthric speech towards enhancing the effectiveness of speech therapy. In *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1000–1005.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551.

Liang, K. J., Li, C., Wang, G., and Carin, L. (2018). Generative Adversarial Network Training is a Continual Learning Problem.

Liu, L., Ling, Z., YuanJian, Zhou, M., and Dai, L. (2018). WaveNet Vocoder with Limited Training Data for Voice Conversion. In *Interspeech 2018*, pages 1983–1987.

Lorenzo-Trueba, J., Yamagishi, J., Toda, T., Saito, D., Villavicencio, F., Kinnunen, T., and Ling, Z. (2018). The Voice Conversion Challenge 2018: Promoting Development of Parallel and Nonparallel Methods.

McFee, B., Lostanlen, V., Metsai, A., McVicar, M., Balke, S., Thomé, C., Raffel, C., Zalkow, F., Malek, A., Dana, Lee, K., Nieto, O., Mason, J., Ellis, D., Battenberg, E., Seyfarth, S., Yamamoto, R., Choi, K., viktorandreevichmorozov, Moore, J., Bittner, R., Hidaka, S., Wei, Z., nullmightybofo, Hereñú, D., Stöter, F.-R., Friesch, P., Weiss, A., Vollrath, M., and Kim, T. (2020). librosa/librosa: 0.8.0.

Mehri, S., Kumar, K., Gulrajani, I., Kumar, R., Jain, S., Sotelo, J., Courville, A., and Bengio, Y. (2016). SampleRNN: An Unconditional End-to-End Neural Audio Generation Model.

Mor, N., Wolf, L., Polyak, A., and Taigman, Y. (2019). A Universal Music Translation Network. In *International Conference on Learning Representations*.

MORISE, M., YOKOMORI, F., and OZAWA, K. (2016). WORLD: A Vocoder-Based High-Quality Speech Synthesis System for Real-Time Applications. *IEICE Transactions on Information and Systems*, E99.D(7):1877–1884.

Motlicek, P., Hermansky, H., Madikeri, S., Prasad, A., and Ganapathy, S. (2020). Am-fm decomposition of speech signal: Application for speech privacy and diagnosis.

Nakashika, T., Takiguchi, T., and Ariki, Y. (2014). Voice Conversion Based on Speaker-Dependent Restricted Boltzmann Machines. *IEICE Transactions on Information and Systems*, E97.D:1403–1410.

Oliphant, T. E. (2006). *A guide to NumPy*, volume 1. Trelgol Publishing USA.

Pasini, M. (2019). MelGAN-VC: Voice Conversion and Audio Style Transfer on arbitrarily long samples using Spectrograms.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pilkington, N., Zen, H., and Gales, M. (2011). Gaussian Process Experts for Voice Conversion. In *Interspeech*, pages 2761–2764.

Qian, K., Jin, Z., Hasegawa-Johnson, M., and Mysore, G. J. (2020). F0-Consistent Many-To-Many Non-Parallel Voice Conversion Via Conditional Autoencoder. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Qian, K., Zhang, Y., Chang, S., Yang, X., and Hasegawa-Johnson, M. (2019). AUTOVC: Zero-Shot Voice Style Transfer with Only Autoencoder Loss.

Ravanelli, M. and Bengio, Y. (2018). Speaker Recognition from Raw Waveform with SincNet.

Ruder, M., Dosovitskiy, A., and Brox, T. (2016). Artistic style transfer for videos. *CoRR*, abs/1604.08610.

Salvador, S. and Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intell. Data Anal.*, 11(5):561–580.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*.

Sisman, B., Yamagishi, J., King, S., and Li, H. (2020). An overview of voice conversion and its challenges: From statistical modeling to deep learning.

Smilkov, D., Thorat, N., Nicholson, C., Reif, E., Viégas, F. B., and Wattenberg, M. (2016). Embedding projector: Interactive visualization and interpretation of embeddings.

Smith, J. O. (2011). *Spectral Audio Signal Processing*. `http://ccrma.stanford.edu/~jos/-sasp/`.

Sotelo, J., Mehri, S., Kumar, K., Santos, J. F., Kastner, K., Courville, A., and Bengio, Y. (2017). Char2wav: End-to-end speech synthesis.

Stylianou, Y., Cappe, O., and Moulines, E. (1998). Continuous probabilistic transform for voice conversion. *IEEE Transactions on Speech and Audio Processing*, 6(2):131–142.

Sun, L., Li, K., Wang, H., Kang, S., and Meng, H. (2016). Phonetic posteriorgrams for many-to-one voice conversion without parallel data training. In *2016 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6.

Sundermann, D., Hoge, H., Bonafonte, A., Ney, H., Black, A., and Narayanan, S. (2006). Text-Independent Voice Conversion Based on Unit Selection. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages I–I.

Taigman, Y., Wolf, L., Polyak, A., and Nachmani, E. (2017). Voice synthesis for in-the-wild speakers via a phonological loop. *CoRR*, abs/1707.06588.

Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J. T., and Ng, R. (2020). Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *arXiv preprint arXiv:2006.10739.*

Tobing, P. L., Wu, Y.-C., Hayashi, T., Kobayashi, K., and Toda, T. (2019). Non-Parallel Voice Conversion with Cyclic Variational Autoencoder.

Toda, T., Chen, L.-H., Saito, D., Villavicencio, F., Wester, M., Wu, Z., and Yamagishi, J. (2016). The Voice Conversion Challenge 2016. In *Interspeech 2016*, pages 1632–1636.

Toda, T., Nakagiri, M., and Shikano, K. (2012). Statistical Voice Conversion Techniques for Body-Conducted Unvoiced Speech Enhancement. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(9):2505–2517.

Turk, O. and Schroder, M. (2010). Evaluation of Expressive Speech Synthesis With Voice Conversion and Copy Resynthesis Techniques. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(5):965–973.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. W., and Kavukcuoglu, K. (2016). WaveNet: A Generative Model for Raw Audio. *CoRR*, abs/1609.03499.

van den Oord, A., Vinyals, O., and Kavukcuoglu, K. (2017). Neural Discrete Representation Learning. *CoRR*, abs/1711.00937.

Van Kuyk, S., Kleijn, W. B., and Hendriks, R. C. (2017). On the information rate of speech communication. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5625–5629.

Veaux, C., Yamagishi, J., and MacDonald, K. (2016). CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit. University of Edinburgh. The Centre for Speech Technology Research (CSTR).

Villavicencio, F. and Bonada, J. (2010). Applying voice conversion to concatenative singing-voice synthesis. In *Interspeech*, pages 2162–2165, Chiba, Japan.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C., Polat, İ., Feng, Y., Moore, E. W., Vand erPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and Contributors, S. . . (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Vásquez-Correa, J. C., Klumpp, P., Orozco-Arroyave, J. R., and Nöth, E. (2019). Phonet: A Tool Based on Gated Recurrent Neural Networks to Extract Phonological Posteriors from Speech. In *Proc. Interspeech 2019*, pages 549–553.

Wan, L., Wang, Q., Papir, A., and Moreno, I. L. (2017). Generalized End-to-End Loss for Speaker Verification.

Wang, X. and Kuo, C. C. J. (1998). An 800 bps VQ-based LPC voice coder. *Acoustical Society of America Journal*, 103(5):2778.

Wang, Y., Skerry-Ryan, R. J., Stanton, D., Wu, Y., Weiss, R. J., Jaitly, N., Yang, Z., Xiao, Y., Chen, Z., Bengio, S., Le, Q. V., Agiomyrgiannakis, Y., Clark, R., and Saurous, R. A. (2017). Tacotron: A Fully End-to-End Text-To-Speech Synthesis Model. *CoRR*, abs/1703.10135.

Watanabe, S., Hori, T., Karita, S., Hayashi, T., Nishitoba, J., Unno, Y., Enrique Yalta Soplin, N., Heymann, J., Wiesner, M., Chen, N., Renduchintala, A., and Ochiai, T. (2018). ESPnet: End-to-End Speech Processing Toolkit. In *Interspeech*, pages 2207–2211.

Wu, Z., Yamagishi, J., Kinnunen, T., Hanilçi, C., Sahidullah, M., Sizov, A., Evans, N., Todisco, M., and Delgado, H. (2017). ASVspoof: The Automatic Speaker Verification Spoofing and Countermeasures Challenge. *IEEE Journal of Selected Topics in Signal Processing*, 11(4):588–604.

Yamamoto, R., Song, E., and Kim, J.-M. (2019). Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks.