

Generative Adversarial Networks for Unsupervised Cross-Domain Speech-to-Speech Synthesis

Mohammad Asif Khan

Matriculation number: 3063566

August 2019

Master Thesis

Computer Science

Supervisors:

Jun.-Prof. Dr. Asja Fischer

Prof. Dr. Jens Lehmann

Dr. Fabien Cardinaux

INSTITUT FÜR INFORMATIK

RHEINISCHE FRIEDRICH-WILHELMS-UNIVERSITÄT BONN

Abstract

The recent advancement in generative adversarial networks provides an end-to-end framework for translating data across domains. The idea has been extensively explored for style transfer in images with an application like translating a photograph to the “Monet Impression, Sunrise”. In this work, we build on this idea and extend it to the task of speech-to-speech synthesis. In our setup, we define speech-to-speech synthesis as the task of translating the voice of the male speaker to that of the female speaker and vice versa. Our work offers several interesting applications. For instance, it can be used to generate artificial data to augment and improve automatic speech recognition systems, or it can be used to achieve better voice privacy in online gaming platforms.

In this thesis, we work with the time-frequency (TF) representation of speech signals. The speech signals are highly periodic and have irregularities and discontinuities. Thus, to obtain the stable TF representation, short-time Fourier transform (STFT) is performed with the overlapping time frames. A key hurdle in using this representation for training neural networks is that the chain of non-linear transformations ignores the overlap factor. As a consequence, the generated TF representations can be unstable which makes it difficult to find their corresponding time-domain waveforms to perform speech synthesis. In such a scenario, the reconstruction of the time domain waveform relies on the consistency¹ of the TF representation.

The TF representation consists of two parts: magnitude and phase spectrogram. We propose two experimental setups: (i) Due to convoluted nature of phase spectrogram, we only use the magnitude part for the training of our model and to perform synthesis at inference time, we reconstruct time-domain speech waveform using the Griffin-Lim algorithm (GLA). In this case, we use the consistency condition of the magnitude spectrogram to constrain our model. (ii) We use the full TF representation. We transform the phase spectrogram to the instantaneous frequency representation which also relates to perceptual quality of sound. In this case, we exploit the consistency condition of the complete TF representation and to perform synthesis at inference we avoid use of GLA and reconstruct time-domain speech waveform using inverse-STFT operation. Our results demonstrate that the consistency constrained model provides a perceptually better quality of speech-to-speech synthesis.

¹The time-frequency representation is said to be consistent if it can be obtained by taking the STFT of any time-domain waveform.

Declaration of Authorship

I, Mohammad Asif Khan, declare that this thesis, titled “Generative Adversarial Networks for Unsupervised Cross-Domain Speech-to-Speech Synthesis”, and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at Rheinische Friedrich-Wilhelms-Universität Bonn.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at Rheinische Friedrich-Wilhelms-Universität Bonn or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. Except for such quotations, this thesis is entirely my own work. I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

Acknowledgement

I am thankful to many people who supported and motivated me in writing this thesis.

First and foremost, I would like to express my profound gratitude towards my supervisors Jun.-Prof. Dr. Asja Fischer and Prof. Dr. Jens Lehmann for their continuous support and supervision. The regular discussion and feedback from Prof. Asja gave me a better understanding of the underlying concepts and helped me in developing critical research skills. I am grateful for all the time she spent in reviewing my work and helping me in writing a better version of the thesis. Without her support, this work would never be possible. I also owe a great many thanks to Prof. Dr. Jens Lehmann. His invaluable advice on various occasions and immense knowledge helped me throughout my master studies. He gave me an opportunity to join his research group at Smart Data Analytics (SDA) which incited me to widen my research horizon. Prof. Jens also trusted me with the responsibility of tutoring in the Knowledge Graph Analysis course which helped me in sharpening my skills. I am deeply grateful for all his support. I could not have imagined having better advisors for my thesis work.

Secondly, I would like to express my warmest gratitude to my co-supervisor Dr. Fabien Cardinaux. I am thankful to Dr. Fabien for hosting me in a vibrant research environment at Sound and Speech Group (SSG), Sony and suggesting this topic. His comments and regular discussions have been critical for my work and have helped me in the timely completion of the thesis. I am also thankful to Dr. Stefan Uhlich and Dr. March Ferras Font for their regular feedback and helpful discussions. A special thanks to the SSG group for providing valuable computational resources which made this work possible.

I am also thankful to my friend Patrick for constructive feedback in improving this thesis.

Lastly, I would like to thank my parents for their unconditional support and everlasting love. My parents never went to school but with their hard work and struggle made it possible for me to pursue the finest level of education. Their inspiration always kept me going, I am deeply grateful and indebted to them.

Table of contents

Abstract	i
Declaration of Authorship	ii
Acknowledgement	iii
Table of contents	iv
List of Figures	vi
List of Tables	vii
1 Introduction	1
1.1 Thesis Outline	5
2 Background	7
2.1 Audio Signal Processing	7
2.1.1 Speech Processing	8
2.1.2 Time Frequency Representation	8
2.1.3 Phase Reconstruction	11
2.2 Deep Generative Models	13
2.2.1 A short introduction to Deep Learning	14
2.2.2 Generative Models	19
2.2.3 Generative Adversarial Networks	20
2.2.4 How to train a GAN?	20
2.2.5 GAN: An Information Theoretic Perspective	21
2.2.6 Evaluation of GANs	24
2.2.7 Variational Auto-encoder	25
2.2.8 CycleGAN	26
2.2.9 Unsupervised Image-to-Image Translation	27
2.3 Deep Generative Models for Audio Signals	28
3 Cross-domain Speech-to-Speech Synthesis	30
3.1 Speech-to-Speech Synthesis	30
3.1.1 Problem definition	31

3.1.2	Magnitude Spectrogram Generation	31
3.1.3	Full Spectrogram Generation	33
3.2	Experimental Setup	35
3.2.1	Dataset	35
3.2.2	Preprocessing	36
3.2.3	Network Details	37
3.2.4	Training Procedure	38
3.2.5	Postprocessing	39
4	Evaluation and Discussion	40
4.1	Evaluation Setup	40
4.1.1	Speaker Identification Network	40
4.1.2	Domain Classification	42
4.1.3	Evaluation Measures	44
4.1.4	Robustness of FID and IS	45
4.2	Results and Discussion	47
4.2.1	Magnitude spectrogram generation	47
4.2.2	Full spectrogram generation	51
4.2.3	Human Evaluation	51
5	Conclusion and Future Research	54
	Bibliography	56

List of Figures

1.1	An example of image to image translation.	2
1.2	Phase and IF representation of a speech signal	4
2.1	An example of DFT of a signal	9
2.2	Time frequency representation of signal	11
2.3	Spectrogram consistency	12
2.4	Maximum Likelihood Principle	20
2.5	A geometric illustration of GAN training.	23
2.6	A geometric view of cycleGAN	27
2.7	Illustration of UNIT Framework	29
4.1	Loss plot of evaluation models (a)	42
4.2	Loss plot of evaluation models (b)	43
4.3	Perceptual evaluation interface	46
4.4	Examples of spectrogram translations.	49
4.5	Mean opinion score	52

List of Tables

3.1	Data Statistics of LibriSpeech Corpus	36
3.2	STFT Configuration	37
3.3	Network time and parameters comparison	38
3.4	Hyperparameter Configuration	38
4.1	Evaluation of domain classifier and speaker identification network. . .	43
4.2	Empirical bound on FID and IS	45
4.3	Robustness of the FID and IS score to Gaussian Noise	46
4.4	Robustness of the FID and IS score to salt and pepper noise	47
4.5	Evaluation of generated samples of different GANs trained on Mag- nitude spectrogram.	48
4.6	Ablation study: Domain accuracy	49
4.7	Ablation study: FID	50
4.8	Evaluation of generated samples of different GANs trained on the full-spectrogram.	51
4.9	Mean Opinion Score	52

List of Algorithms

2.1	Griffin-Lim Algorithm	13
2.2	Stochastic Gradient Descent	18
2.3	Minibatch training of a GAN	21

Chapter 1

Introduction

In past two decades, deep learning (DL) research has gained a lot of popularity and has been widely used to solve various problems in artificial intelligence (AI) like object recognition in computer vision, sentiment classification in natural language processing (NLP), etc. The majority of success in DL research is in the discriminative class of models, where a classifier is trained to map the input data to its label class. The label class can be a discrete number for classification or a continuous value for regression. A particular class of emerging deep learning research focuses on generative models (GMs) where the objective is to estimate the joint probability distribution of the input data and class labels. In recent years GMs have attracted a lot of interest mainly due to advancement in deep learning methods and computational resources. [Goodfellow et al. \[2014\]](#) proposed generative adversarial networks (GANs) which efficiently learn to generate samples from high dimensional probability distributions in an adversarial setting. GANs have achieved state-of-the-art performance for various computer vision tasks. For instance, generating high-resolution images ([Karras et al. \[2017\]](#)) and multi-stage generation ([Denton et al. \[2015\]](#), [Karras et al. \[2017\]](#)). However, their application for audio or speech data is still limited.

A problem often encountered in developing ML systems is the limited amount of data at the training time. This results in low performance when evaluated on samples different from those in the training set. The low diversity of samples at training time could also lead to poor generalization performance. Thus, for a robust model, one of the requirement is that the training samples should reflect the true population of samples in the data distribution. For example, to detect an image of a cat we need to show the model images of different kinds of cats during training. This allows the model to generalize the notion of a cat. An interesting application of GANs is to use them for generating high-fidelity data which then can be used to augment the training of ML systems.

GANs learn to generate data by applying a parametric transformation on a noise distribution. One limitation in this setup is of generating data with some predefined attributes. For example, generating an image of a car given we define the shape of a car. An interesting class of GANs ([Zhu et al. \[2017\]](#), [Liu et al. \[2017\]](#)) addresses



Figure 1.1: An example of image to image translation. Rows from top to bottom are tasks: day to night, sunny to rainy, summer to snowy, winter to summer and vice versa. The first and third column from the left are examples of input domain and second and fourth are example of translated images. Taken from [Liu et al. \[2017\]](#)

this issue by posing it as a translation problem from data of one domain to that of another domain. The main idea in this setup is to learn specific characteristics or features of an image in the source domain and then use them to generate an image in the target domain. In previous example, shape of the car can be one domain and image as the other domain. Figure 1.1 further gives an example of translating images from a given source domain to the desired target domain.

DL models have already offered a significant promise in building robust speech detection systems. However, such systems are usually limited and perform poorly when evaluated on unseen data. For example, consider speech assistant systems which are trained on the task of transcribing speech to text. Evaluation of such systems on an unseen set of speakers often fails. This is mainly due to the limited diversity of speakers in the training data. A robust system should learn the high variability of the features like phonetics of a speaker, accent and dialect associated with a language, etc. This requirement is often difficult to fulfill due to constraints imposed by the data collection process. Thus, an unsupervised framework to generate speech of the target domain is of great help for the development of robust systems. Furthermore, the existence of such a system can be used for voice privacy in online gaming platforms. The players can choose to be anonymous by translating their voice to an arbitrary choice of voice.

In this thesis, following the work of [Liu et al. \[2017\]](#), we propose a GAN frame-

work for unsupervised speech-to-speech synthesis. In our work, we define speech-to-speech synthesis as the task of translating the voice of male speakers to female speakers and vice versa. Since our framework is completely unsupervised it requires no pairing of samples of male and female speakers for training.

Speech signals are slow time-varying signals represented as a waveform of time-amplitude variations. Oord et al. [2016] proposed an autoregressive model to generate high-quality speech signals from the time domain representation. However, the approach is limited and results in a slow real-time inference. This is mainly attributed to the high sampling rate of speech data which makes it computationally difficult to generate speech waveform in real-time. An alternative to the time-amplitude representation is the time-frequency representation where a signal is expressed in terms of the energy of its frequency components. Recently, several GAN based approaches (Engel et al. [2019], Marafioti et al. [2019]) have been proposed utilizing the time-frequency representation for generating high-quality audio data, which significantly outperformed networks trained on the time domain representation. Therefore in this work, we use the time-frequency representation of speech.

The time-frequency representation comprises of two parts: magnitude spectrogram and phase spectrogram. In Section 2.1.2 we will discuss the short-time Fourier transform (STFT) operation used to obtain the time-frequency representation. The highly periodic nature of speech signals results in irregularities and discontinuities in the respective waveform. The human perception is sensitive to such discontinuities and irregularities. As a result, maintaining such irregularities in the time-frequency representation is crucial. The STFT operation is performed with overlapping time frames. As a consequence, the magnitude spectrogram is well structured in the form of clearly visible contour regions and formant trajectories which are sensitive to irregularities in the waveform. A specialist in audio signals can read this representation and easily associate it with its perceptual quality. However, a phase spectrogram has jump discontinuities at various periodic components of a signal. Thus, the visual interpretation of the phase is not clear and it is difficult to derive a proper understanding of the nature of the signal from it. Due to these reasons, the magnitude representation is considered as the natural choice in ML applications.

The STFT operation is performed with overlapping time frames which gives an over-complete time-frequency representation. In general, for an arbitrary time-domain signal by performing STFT operation followed with inverse STFT operation we can reconstruct the same time-domain waveform. However, in applications where we discard phase spectrogram and apply an arbitrary linear or non-linear transformation on the magnitude spectrogram, we have no guarantee of reconstructing a corresponding time-domain waveform from modified magnitude spectrogram. To reconstruct a coherent waveform in the time domain requires the right combination of magnitude and phase spectrogram. Since there are infinitely many combinations, the reconstruction of the time domain speech waveform from the magnitude spectrogram is an ill-posed problem (Jaganathan et al. [2015]).

There are iterative methods like the Griffin Lim (Griffin and Lim [1984]) algorithm to find the closest time domain waveform from a magnitude spectrogram.

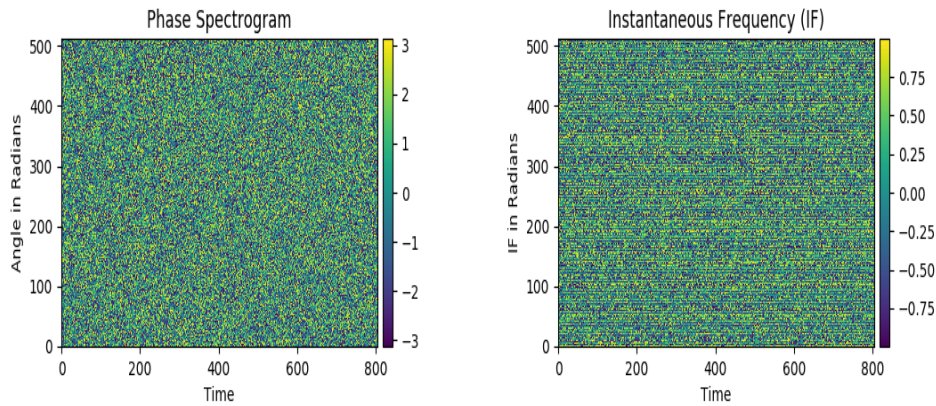


Figure 1.2: The time-phase representation (left) and the instantaneous frequency (IF) representation (right) of the signal illustrated in Figure 2.1.

But such approaches often add artifacts and require a large number of iterations to converge to the solution unless the spectrogram is consistent. In Section 2.1.3 we discuss in detail the condition of spectrogram consistency. An alternative to iterative methods is to utilize the instantaneous frequency (IF) representation of the phase. The IF is obtained by first unwrapping the phase, by adding 2π whenever there is a phase discontinuity and then taking finite-difference along the time axis. Unwrapping makes the phase grow linearly and the finite-difference captures signal oscillations resulting in a solid rainbow-like pattern. This representation also relates to the perceptual quality of harmonics present in the waveform. Since the phase-to-IF conversion is an invertible transformation we can fully reconstruct the phase spectrogram from IF. Figure 1.2 gives a comparison between phase and IF representations of a speech signal. By combining magnitude spectrogram and IF we can train a network to generate the full spectrogram representation and thus avoid the use of an iterative algorithm. However, the consistency problem persists. As the network can generate a spectrogram of any arbitrary overlap factor different from the one used to obtain real spectrogram. As a consequence, the generated spectrogram might not necessarily correspond to any valid time-domain waveform.

In this work, we present two frameworks of GANs for unsupervised cross-domain speech-to-speech synthesis. In the first framework, we use the magnitude spectrogram of a speech signal and use iterative algorithms for waveform reconstruction. Since the consistency of the spectrogram is a critical requirement to obtain a valid time-domain waveform. We define a spectrogram consistency loss to constrain the generator. In the second framework, we utilize the complete time-frequency representation by combining the magnitude spectrogram and the IF representation of the phase spectrogram. Thus we avoid the use of iterative methods for signal reconstruction and we use the consistency condition of the complete time-frequency representation to define a spectrogram consistency loss term for the generator networks. We discuss the details in Section 3.1.3.

Our results show the effectiveness of the approach for the cross-domain speech-to-speech synthesis. To evaluate the quality and diversity of generated samples we compute the inception score (IS), the Fréchet inception distance (FID) and domain accuracy. The IS and FID require the feature representation of the spectrogram. We train a time-delay neural network (TDNN) model for the speaker identification task and later use it as a feature extractor. For domain accuracy, we train a binary classifier to discriminate between the voice of male and female speakers. We then compare the accuracy of the classifier on the true set of spectrograms and on the generated set of spectrograms. Furthermore, we also perform the qualitative evaluation of generated speech samples. We performed two perceptual tests to evaluate the quality of speech of generated samples and the extent to which generated samples sounds like a female or a male voice.

We summarize the main contribution of our work as follows:

- (i) Our network inspired by the UNIT architecture works on the shared latent space assumption. To our knowledge, this setup has not been used previously for speech-to-speech synthesis.
- (ii) We impose the spectrogram consistency constraint on our cross-domain speech-to-speech synthesis network. This allows faster convergence of iterative algorithms and provides a coherent time-domain waveform reconstruction.
- (iii) We train our framework on the LibriSpeech (Panayotov et al. [2015]) corpus of 251 speakers and evaluate it on the completely unseen set of 40 speakers.
- (iv) We propose to generate the full spectrogram representation where we use the IF representation of the phase along with the magnitude spectrogram. Furthermore, we propose a loss term exploiting consistency of full spectrogram.

Our framework was effective at the task of voice conversion. We achieved quantitative as well as qualitative improvements by imposing the spectrum consistency constraint. In the case of a full time-frequency representation, we observed that the network added artifacts like, echoing in generated speech samples. We attribute this to the different nature of magnitude and phase spectrogram. The magnitude representation is sparse, and phase representation is dense, and combining them as two channels makes it difficult for the network. We think it can be improved by performing extensive hyperparameter tuning, especially hyperparameter of the magnitude and phase term in the loss function as well as of the spectrogram consistency term. Due to limited time and high computational cost of our framework, we left this as future work.

1.1 Thesis Outline

We structure the thesis as follows. In Chapter 2 we review concepts of signal processing, DL and GMs specifically required for the understanding of our work. In Chapter 3 we formally define the problem statement and discuss the formulation of

our framework and training setup. In Chapter 4 we explain our evaluation setup, report results along with an extensive discussion of various setups. Finally, we conclude our work in Chapter 5 along with the discussion on the prospects of this work.

Chapter 2

Background

In this thesis, we work with methods from signal processing and machine learning. In this chapter, we first start with a brief introduction to audio signal processing in Section 2.1 and build the foundation for the later part of the thesis. Later we follow our discussion and build concepts of deep learning with a particular focus on deep generative models in Section 2.2.

2.1 Audio Signal Processing

Audio signal processing (ASP) deals with the perception of sound to human auditory system. The physical form of sound or an audio signal can be seen as an encoding of air pressure over time. Formally, an audio signal is a function $x : \mathbb{R} \rightarrow \mathbb{R}$, that maps every point $t \in \mathbb{R}$ in time to sound pressure value $x(t) \in \mathbb{R}$. Such a representation is called analog or continuous time (CT) signal. When dealing with digital systems a critical step called digitization is used to process and store audio signals. Digitization is a two step process. The first step involves discretization of the domain of a signal called sampling. The second step involves the approximation of the range of a signal called quantization.

Sampling is a procedure to transform a CT signal to discrete-time (DT) signal. One common way to implement it is known as equidistant sampling. In an equidistant sampling, for a given CT signal $x : \mathbb{R} \rightarrow \mathbb{R}$ and a sampling constant $T > 0$, a DT signal $x : \mathbb{Z} \rightarrow \mathbb{R}$ is derived as:

$$x[n] = f(n \cdot T) \tag{2.1}$$

for all $n \in \mathbb{Z}$. In this sampling scheme, any two neighboring points are separated by an equal distance of T . The sampling rate of a signal is defined as the number of samples per second given by $F_s = 1/T$ and is measured in Hertz (Hz). Sampling, in general, is a non-invertible transformation and results in the loss of information. However, for a bandlimited signal with a maximum frequency of f_m sampling is invertible under Nyquist criterion that is $F_s \geq f_m$. The condition is known as **Nyquist Sampling Theorem** (Nyquist [1924]).

Quantization is a step to discretize real amplitude values using a finite set of integer values. The general process involves the partitioning of the amplitude range in disjoint sets and using a codebook to choose appropriate code value. A signal after digitization is an integer-valued function $F : \mathbb{Z} \rightarrow \mathbb{Z}$.

2.1.1 Speech Processing

Speech processing is the application of ASP to the understanding and analysis of speech signals. The speech signal is described as a composition of sequences of sounds produced by acoustical excitation of the vocal tract. The vocal tract vibrates at a particular frequency which is called fundamental frequency of the sound. The fundamental frequency of male speakers is generally in the range of 50-200 Hz, of female speakers in the range of 150-300 Hz and children in the range 200-400 Hz. Figure 2.1 (left side) gives an example of a speech signal of 10 sec. sampled at 16kHz.

The speech signals are also referred to as a slow time-varying signal i.e. in a very short duration of time in 30-60 ms the characteristics of speech signals are stationary. To understand the acoustic properties of speech a common step involves decomposing it into fundamental components known as frequencies.

In this thesis, we work with the spectrogram representation of speech signals. In the following section, we will discuss the Fourier transform (FT) and other methods for time-frequency representation. Since in this thesis, we work with DT signals we will limit our discussion to the discrete case of FT.

2.1.2 Time Frequency Representation

The Fourier transform (FT) is the widely used method of understanding the time-frequency dynamics of a signal. It maps a time-domain signal to a frequency domain which provides an insight to the spectrum of frequency components composing the signal. In the book, ‘The World According to Wavelets’, Hubbard [1998] gives the following elegant characterization of FT:

The Fourier transform is the mathematical procedure that breaks up a function into the frequencies that compose it, as a prism breaks up light into colors. It transforms a function f that depends on time into a new function, $\hat{f}$, which depends on frequency. This new function is called the Fourier transform of the original function (or, when the original function is periodic, its Fourier series).

Formally, the FT of a DT signal $x \in \mathbb{R}^L$ is defined as:

$$\hat{x}(k) = \sum_{n=0}^{L-1} x(n) \cdot e^{-2\pi n k j / L} \quad (2.2)$$

where L is the length of signal, $k \in \{0, \dots, L-1\}$ and $e^{-2\pi n k j / L}$ are the complex frequency components in $x(k)$. Each entry $\hat{x}(k)$ contains information about the magnitude as well as the phase of the complex frequency component $e^{-2\pi n k j / L}$. The FT is an invertible transformation and we can obtain the time domain signal back

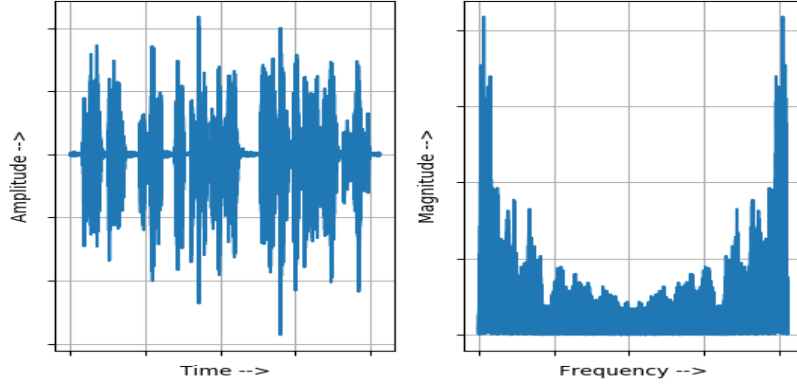


Figure 2.1: An example of time domain signal and its Fourier transformation.

from the frequency domain by summing up the frequency component. Formally, an inverse FT of a signal $x \in \mathbb{C}^L$ is defined as:

$$x(n) = \sum_{k=0}^{L-1} \hat{x}(k) \cdot e^{2\pi n k j / L} \quad (2.3)$$

In practice, FT is computed using fast Fourier transform (FFT) algorithm ([Heideman et al. \[1984\]](#)). We avoid the details in our discussion. The FT provides information about the frequency components of a whole signal. However, it fails to provide information about the temporal onset of frequencies and thus is not helpful for the localization of frequencies i.e. for analyzing which frequencies are present at any given point of time. In terms of speech signals, FT gives information about average sound events present in a signal but provides no information about which sound event occurred at a given time. Figure 2.1 gives an example of an FT of a speech signal. We can see it gives the overall measure of frequency but provides no information about the time location. To understand the nature and onset of frequencies we need a transformation which not only provides the spectral components but also provides the temporal properties of such components. To address this problem of time-frequency localization, [Gabor \[1946\]](#) proposed a modified version of FT commonly referred to as short-time Fourier transform (STFT).

2.1.2.1 Short Time Fourier Transform

The STFT of a time-frequency representation of a signal, in general, can be thought of as a compromise between a time and a frequency representation. It provides a way to simultaneously represent time as well as frequency components of a signal thereby providing a way to locate frequency events over time. The key idea behind STFT is to apply the FT locally over a signal using a window function. The window function is defined over an interval and provides a way to analyze the frequency content of a signal in a given segment.

Formally, the STFT of a signal $x \in \mathbb{R}^L$ with the window $w \in \mathbb{R}^L$, hop length $h \in \mathbb{N}$ and frequency steps $M \in \mathbb{N}$ is defined as:

$$\hat{X}_w(a, b) = \sum_{l=0}^{L-1} x(l) \cdot w(l - nh) e^{-2\pi j ab/M} \quad (2.4)$$

for all $a \in \{0, \dots, N-1\}$ and $b \in \{0, \dots, M-1\}$ where N is the number of time steps and M is the frequency resolution. The hop length h is a measure of redundancy, it is defined as the number of time steps between two consecutive window frames. Depending on the sampling frequency and hop length we can also express frequency step a in Hz as $\tilde{a} = a \cdot F_s/N$ and time step b in seconds as $\tilde{b} = b \cdot h/F_s$, where F_s is the sampling rate of signal x . The entry $X(a, b)$ represents the Fourier coefficient present at the a^{th} frequency and b^{th} time step.

Similar to FT we can also define the inverse transformation for STFT. For a given STFT matrix $\hat{X}_w$ under $\hat{w}$ an inverse short-time Fourier transform (ISTFT) is defined as:

$$x(n) = \sum_{a=0}^{N-1} \sum_{b=0}^{M-1} \hat{X}_w(a, b) \cdot \hat{w}(n - ha) e^{2\pi j ab/M} \quad (2.5)$$

where $\hat{w}$ is an FT representation of window w . The window size controls the time-frequency resolution. A smaller window size in the time domain gives a good time resolution but a bad frequency resolution. Likewise, a large window size gives a better frequency resolution but a bad time resolution. The simultaneous time-frequency analysis is governed by the Heisenberg Uncertainty principle (Gabor [1946]). Improving the resolution in one domain affects the quality in the other domain and vice versa. Fortunately, the Gaussian window has minimum uncertainty and is a preferred choice for STFT. In this work, we use a type of Gaussian window called “von Hann” window originally proposed by Harris [1978].

The STFT matrix has two parts: the magnitude and the phase information. Due to complexities in dealing with the phase information discussed in a later part in Section 2.1.3, the magnitude information is a preferable representation for machine learning tasks. The magnitude of an STFT matrix $\hat{X}_w(a, b)$ is defined as $|\hat{X}_w(a, b)|$. In Figure 2.2 (left side) gives an example of an STFT of a signal with a Gaussian window. The colored region in the spectrogram reflects the magnitude of the coefficients. A log transformation defined as $\ln(1 + |\hat{X}_w(a, b)|)$ is commonly applied on the magnitude spectrogram to enhance the low magnitude coefficients. In Figure 2.2 the middle part gives an example of the log transformation.

2.1.2.2 Mel Spectrogram

Human ears perceive the perceptual difference in sound quality based on the pitch of a signal. The pitch is a function of frequency and can be interpreted as the perceived frequency. The mel scale introduced by Stevens and Volkmann [1940] is a transformation which relates the real frequency to the perceived frequency also

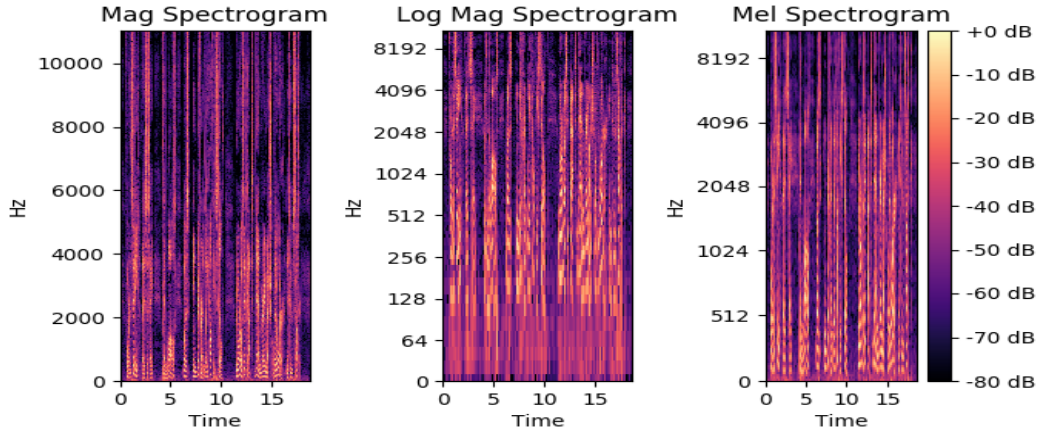


Figure 2.2: Time frequency representation of signal illustrated in Figure 2.1. On the left is the magnitude spectrogram, in the middle is the log magnitude spectrogram and on the right is the mel spectrogram. For this example we use 1024 point FFT with 75% overlap, hanning window of size 1024 and 128 mel filters.

known as pitch. Various forms of mel functions have been proposed. One commonly used transformation is given by

$$m = 2595 \log\left(1 + \frac{f}{700}\right) \quad (2.6)$$

mapping the frequency f (in Hz) to m mels. The magnitude spectrogram consists of different frequency bands. To perceive harmonics of each band it is essential that all bands have an equal resolution on the mel scale. This is achieved by applying a bank of overlapping triangular filters around the center frequency of each band. The Mel spectrogram is obtained by scaling the magnitude spectrogram to the mel scale and then passing it through the filterbanks. For further details on mel filterbanks, we refer to [Kopparapu and Laxminarayana \[2010\]](#). Figure 2.2 gives an example of a mel spectrogram and its comparison with the magnitude and log magnitude representation.

2.1.3 Phase Reconstruction

The STFT representation $\hat{X}$ is a matrix of complex numbers. The squared magnitude of $\hat{X}$ is interpreted as an energy spectrum of a signal where each entry (a, b) represents the energy of the signal at the frequency step a and time step b . The angle of the STFT matrix $\hat{X}$ represents a phase of a signal at the frequency step a and time step b . The phase representation consists of discontinuities at periodic components of a signal and it is not clear how this representation relates to the perceptual quality of sound. Thus, using the phase spectrogram increases the complexity of the analysis. As a consequence, the magnitude information of the STFT matrix is a preferable choice for various signal processing and machine learning tasks.

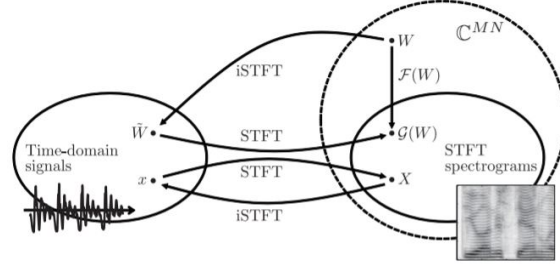


Figure 2.3: An example of spectrogram consistency. Taken from [Le Roux et al. \[2010\]](#)

One problem with this choice is the consistency of the spectrogram under arbitrary transformations. The windowing in STFT is done with overlapping time frames. To preserve time-frequency dynamics any transformation on STFT should also take into account the proportion of the overlap. However, the methods in signal processing, as well as ML, often ignore this information. As a result, the modified magnitude spectrogram might not even have a valid time-domain reconstruction. Thus, to reconstruct a signal a crucial requirement is a right combination of phase information which in turn depends on the consistency of the spectrogram representation.

2.1.3.1 Spectrogram Consistency

As discussed previously any transformation applied to the magnitude spectrogram makes it difficult to reconstruct the time domain signal. In this work, we generate the spectrogram using a GAN. The details are discussed in the latter part of the thesis in Chapter 3. One central problem underlying the generation is the consistency of the spectrogram, that is to say, finding a unique time-domain signal from a generated spectrogram representation. In this part, we formally define the condition for consistency of the spectrogram.

Let $\mathbb{C}^{MN}$ be a space of complex matrices, $\mathbb{H}^L$ be a space of time-domain signals and $\mathbb{V}^{MN}$ be a space of consistent spectrograms. A consistent spectrogram $X \in \mathbb{V}^{MN}$ for some $x \in \mathbb{H}^L$ is a matrix which can be obtained by applying STFT on x . Since, $\mathbb{V}^{MN} \subset \mathbb{C}^{MN}$ implies $\exists W \in \mathbb{C}^{MN}$ such that there is no unique reconstruction of W in $\mathbb{H}^L$. Thus for any $W \in \mathbb{C}^{MN}$ to be a consistent spectrogram it should be in a null space of a linear transformation $\mathcal{F}$ defined as:

$$\mathcal{F}(W) = \text{STFT}(\text{ISTFT}(W)) - W \quad (2.7)$$

The Figure 2.3 gives the geometrical illustration of the consistency of a spectrogram.

2.1.3.2 Griffin Lim Reconstruction

The Griffin Lim algorithm (GLA) proposed by [Griffin and Lim \[1984\]](#) makes use of the consistency constraint to reconstruct the closest time-domain signal from a modified magnitude spectrogram. The GLA finds the phase for a magnitude spectrogram by projecting it into the null space of the operator $\mathcal{F}(W)$. Thus, it minimizes the projection error of Equation 2.7 by iteratively projecting the signal across two domains i.e. from frequency to time and time to frequency. Under sufficient overlap of the window in the STFT operation, the GLA guarantees to reconstruct a solution approximating the corresponding time-domain waveform. Algorithm 2.1 further defines the GLA.

Algorithm 2.1 Griffin-Lim Algorithm

```

1: procedure GLA( $\hat{\mathcal{M}}, n$ )           ▷ Input: Modified Magnitude Spectrogram and
   number of training iterations
2:    $\phi_0 \leftarrow \angle \mathcal{N}(0, I)$            ▷ Initial random phase in radians
3:    $\hat{\mathcal{S}}_0 = \hat{\mathcal{M}} \cdot e^{-j\phi_0}$            ▷ initial full spectrogram
4:   for i in range (1,n) do
5:      $\hat{\mathcal{S}}_i = \text{STFT} \circ \text{ISTFT}(\hat{\mathcal{S}}_{i-1})$ 
6:   end for
7:    $\mathbf{x} = \text{ISTFT}\{\hat{\mathcal{S}}_n\}$ 
8:   return  $\mathbf{x}$ 
9: end procedure

```

One problem often encountered with GLA is the low-quality reconstruction when dealing with high-quality sound signals. Many improvements of the GLA have been proposed. [Perraudin et al. \[2013\]](#) formulate the phase reconstruction as an optimization problem and modify the conventional GLA. ([Prša et al. \[2017\]](#)) utilized the phase magnitude relationship proposed by ([Auger et al. \[2012\]](#)) and proposed a stable way to integrate the magnitude spectrogram at different scales. In this work, we limit our discussion to the GLA. For further details we refer to ([Perraudin et al. \[2013\]](#), [Le Roux et al. \[2010\]](#), [Jaganathan et al. \[2015\]](#)).

2.2 Deep Generative Models

In this section, we build our discussion around deep generative models. We will start with a gentle introduction to deep learning in Section 2.2.1. We will then briefly discuss about generative models in Section 2.2.2 followed with a detailed discussion on generative adversarial networks (GANs) in Section 2.2.3. Further, in Section 2.2.4 we look at the training procedure of GAN, its convergence properties in Section 2.2.5 and evaluation measures in Section 2.2.6. Finally, we conclude our discussion with other GMs, VAE in Section 2.2.7, cycleGAN in Section 2.2.8 and unsupervised image-to-image translation in Section 2.2.9.

2.2.1 A short introduction to Deep Learning

In recent years deep learning (DL) has emerged as a popular choice for modeling various artificial intelligence (AI) tasks ranging from object recognition in computer vision to language models in NLP. DL works on the principle of building a solution to a complex problem by decomposing it into simpler ones which reduce the complexity of the problem and provides insights to model non-linear dynamics of data.

Goodfellow et al. [2016] outline DL as: “A way to allow computers to learn from experience and understand the world in terms of a hierarchy of concepts, with each concept defined through its relation to simpler concepts. By gathering knowledge from experience, this approach avoids the need for human operators to formally specify all the knowledge that the computer needs. The hierarchy of concepts enables the computer to learn complicated concepts by building them out of simpler ones. If we draw a graph showing how these concept defined as the decomposition of complex concepts into simple ones and the recombination into new complex concepts. An algorithm therefore has to establish a hierarchy of concepts. A visualization of this hierarchy would be a multi-layered graph, that can be called deep in a graph theory context.”

2.2.1.1 Artificial Neural Network

An artificial neural network (ANN) is a learning model inspired by the working of biological neurons. It consists of an arbitrary number of small computational units called artificial neurons and solves a learning problem by modeling the strength of the connection between them. This nature of ANNs allows them to model functions of arbitrary complexity and thus classifies them as universal approximators (Hornik et al. [1989]).

We start our discussion with the basic form of an ANN commonly referred as *perceptron*. The *perceptron* uses a single neuron to approximate a function. Following Mitchell et al. [1990] we define the perceptron model. For an input $x \in \mathbb{R}^D$ and the parameterized function $f : x \rightarrow y$ defined as $f(x; w, b) = w^T x + b$ where $y \in \mathbb{R}$ is an output score, $w \in \mathbb{R}^D$ is a weight parameter and $b \in \mathbb{R}$ is a bias term. For a binary classification task the output $f(x; w, b)$ is further passed through the decision function also referred as *activation function* formally given by Equation 2.8.

$$\hat{y} = \begin{cases} 1, & \text{if } f(x; w, b) > 0 \\ 0, & \text{otherwise} \end{cases} \quad (2.8)$$

The initial parameters (w, b) are set to zero. At every step of learning the output, y is computed and the parameters are adjusted based on the mean-squared-error between target and predicted output. One major limitation of the *perceptron* is that it cannot model non-linear relations, as for instance, XOR function.

The *Multilayer perceptron* (MLP) is a class of ANNs which unlike the perceptron uses multiple neurons over several layers to approximate the unknown function. The interconnected neurons form a hierarchical network which allows to model a function

of arbitrary complexity. Also, there are various families of activation functions which can further improve the learning process.

Formally, a MLP can be considered as a chain of functional maps applied on data to unravel the non-linearities over several layers of the hierarchy. Let L be the number of layers, $x \in \mathbb{R}^d$ be an input representation, for a layer l : $d_l \in \mathbb{N}$ be number of neurons, v^l be an input representation, u^l be an output representation, $f_l : \mathbb{R}^{d_{l-1}} \rightarrow \mathbb{R}^{d_l}$ be a transformation mapping u^{l-1} to u^l , let $\theta^{l-1,l} = \{w^{l-1,l}, b^{l-1,l}\}$ be parameters connecting layer $l-1$ to l and σ^l be an activation function, then the output of the MLP $f(x)$ is defined as:

$$\begin{aligned} v^0(x) &= x \\ v^l(x) &= u^{l-1}(x) \\ u^l(x) &= \sigma^l(f^l(v^l, \theta^{l-1,l})) \\ f^l(v^l, \theta^{l-1,l}) &= w^{l-1,l} \cdot u^l + b^{l-1,l} \\ f(x) &= u^L \circ u^{L-1} \circ \dots \circ u^1 \circ u^0(x) \end{aligned} \tag{2.9}$$

The MLPs are efficient in modeling any class of functions. However, it is difficult to train MLPs with a large number of neurons in the hidden layers. This is because each neuron connects to all the neurons of the previous layer. As a result the number of parameters increase significantly making learning infeasible. Another issue with MLPs is that they are permutation invariant i.e. if the feature space is shuffled the output of the network is still the same. As a result, it fails in modeling the spatial correlation in data. For further details we recommend readers to refer to [Goodfellow et al. \[2016\]](#).

2.2.1.2 Convolutional Neural Network

Convolutional neural networks (CNN) ([LeCun et al. \[1995\]](#)) take the inspiration from the receptive-field of the visual cortex. In the visual cortex, neurons form local regions from receptive-field a.k.a feature-maps where neurons respond selectively to various attributes of visual input. The CNN exploits this idea to implement local feature-maps which by weight sharing learns different levels of abstraction from a given input.

A convolution is a linear operator which takes two functions as an input and provides another function as an output. This composition is defined to capture the *cross-correlation* between the two functions. Formally, for an input $x \in \mathbb{R}^N$ and a receptive field in $w \in \mathbb{R}^K$ the convolution operator is defined as:

$$y(n) = (x * w)(n) = \sum_{k=0}^K x(k) \cdot w(n - k) \tag{2.10}$$

where $n \in \{0, \dots, N + K - 1\}$. Likewise, a convolution operator can be extended to multidimensional input $x \in \mathbb{R}^{M_1 \times M_2 \times \dots \times M_d}$. For a more comprehensive explanation we refer to [Goodfellow et al. \[2016\]](#).

The CNN uses a convolutional operator to apply a parametric transformation on the input representation. Thus, learning parameters which are effective in capturing spatial or temporal correlations in the data. For an input $x \in \mathbb{R}^{m \times n \times c}$, the convolutional layer $l \in L$ with filter maps $w \in \mathbb{R}^{k_l \times k_l \times c_{l-1}}$ in CNNs is a three step operation: (i) Apply the convolutional operator $f_l : \mathbb{R}^{m_{l-1} \times n_{l-1} \times c_{l-1}} \rightarrow \mathbb{R}^{m_l \times n_l \times c_l}$ to obtain an intermediate output. (ii) Apply the activation function σ^l and (iii) Apply the pooling operation $g_l^d : \mathbb{R}^{m_{l-1} \times n_{l-1} \times c_{l-1}} \rightarrow \mathbb{R}^{m_{l-1}/d \times n_{l-1}/d \times c_{l-1}}$ to downsample the output of activations, where $c_0 = c$ is number of input channels, $m_l \times n_l \times c_l$ is the shape of the output of l^{th} layer and σ^l is the activation function used in l^{th} layer.

Depending on the design choice, the step (iii) can also be omitted sometimes. An optimal choice of the number of filter maps and its size is a hyperparameter optimization problem and is usually decided by heuristics or grid search. A stack of convolutional layers followed by some layers of an MLP is a preferable design choice of CNNs and have shown a lot of success particularly in computer vision tasks like object recognition (Krizhevsky et al. [2012], Donahue et al. [2014]). However, for tasks like super-resolution or object localization where the output is essentially in a spatial domain a fully convolutional (FCN) approach is a preferred design choice. The FCNs learn a path from pixels in the first layers to the pixel in the deeper layers thus preserving point to point correspondence and produce an output in the spatial domain.

2.2.1.3 Residual Networks

The training of deep neural networks is difficult. The increasing depth results in the saturation of the performance followed with a rapid degradation. The residual networks (ResNets) introduced by He et al. [2016a] address this problem by proposing skip identity connections. The key idea behind ResNets is that for every shallow network there is a corresponding deep network with identity mappings. The ResNets introduce *skip identity connections* to learn a change in a function rather than a function itself which is a relatively simpler learning task (He et al. [2016b]). For an input x , the intermediate output $u^l(x, \theta^l)$ of layer l the output of the residual block is defined as:

$$u^{l+1}(x, \theta^{l+1}) = u^l(x, \theta^l) + \mathcal{F}^l(u^l(x, \theta^l), \theta^{l+1}) \quad (2.11)$$

2.2.1.4 Optimization in Deep Learning

The learning of parameters in NNs is formulated as an optimization problem. In this section we will discuss in the context of supervised learning where we have associated labels for data points. Our discussion follows from Mitchell et al. [1990].

Let $\mathcal{D} = (\mathcal{X}, \mathcal{Y}) = \{(x_1, y_1), \dots, (x_N, y_N)\}$ be a set of N independent and identically distributed (i.i.d) data points sampled from the data probability distribution $P_{(x,y) \sim \mathcal{X}, \mathcal{Y}}(x, y)$, where $x_i \in \mathbb{R}^d$ is a feature representation of i^{th} sample and $y_i \in \mathbb{R}$ is an associated label. We split $\mathcal{D}$ into two sets namely the *training* and the *test* set given as: $\mathcal{D}_{\text{train}} = (\mathcal{X}_{\text{train}}, \mathcal{Y}_{\text{train}}) = \{(x_1, y_1), \dots, (x_M, y_M)\}$, $\mathcal{D}_{\text{test}} = (\mathcal{X}_{\text{test}}, \mathcal{Y}_{\text{test}}) =$

$\{(x_{M+1}, y_{M+1}), \dots, (x_N, y_N)\}$. In supervised learning our objective is to learn a model represented as a parametric function $f : \mathcal{X}_{\text{train}} \rightarrow \mathcal{Y}_{\text{train}}$ and later use it to make prediction on the test set as $f : \mathcal{X}_{\text{test}} \rightarrow \mathcal{Y}_{\text{test}}$. For a parametric function f we define a performance measure a.k.a. loss function $L : \mathcal{X} \times \mathcal{Y} \times \mathbb{R} \rightarrow [0, \infty)$ which gives the estimate of quality of the model prediction $\mathcal{L}(x, y, f(x))$ for a sample $(x, y) \in \mathcal{D}_{\text{train}}$.

The learning can now be interpreted as an optimization problem to learn a function f which minimizes the expected value of the loss function a.k.a. risk of a function f expressed as:

$$R_f = \int_{\mathcal{X} \times \mathcal{Y}} L(x, y, f(x)) p(x, y) dy dx \quad (2.12)$$

For a model to generalize well on unseen samples we need an optimal parametric function $\hat{f}$ which has minimum loss on test points:

$$\hat{f} = \underset{f}{\operatorname{argmin}} \frac{1}{N - M} \sum_{i=M+1}^N L(x_i, y_i, f(x_i)) \quad (2.13)$$

A main problem with this estimate is that we do not know $P_{(x,y) \sim \mathcal{X}, \mathcal{Y}}(x, y)$. However, under an assumption data points are i.i.d., we can alternatively minimize the empirical risk:

$$\hat{R}_f = \frac{1}{M} \sum_{i=0}^M L(x_i, y_i, f(x_i)) \quad (2.14)$$

for a fixed f if $M \rightarrow \infty$ then $\hat{R}_f \rightarrow R_f$. This form of optimization is known as *empirical risk minimization* (ERM).

The optimization of deep learning comes with various challenges: (i) The deep learning optimization problems are solved using gradient descent-based methods. As a result, a certain choice of the loss function is also not feasible. For example, a 0-1 loss used in the classification task is not differentiable. (ii) The ERM principle minimizes the expected value of the training loss. An associated problem is that a model with a sufficiently large number of parameters can memorize the training data with empirical risk $\hat{R}_f \approx 0$ and fail to generalize on the testing data resulting in overfitting. (iii) The learning task can be too complex for a model or a model is not expressive enough due to limited number of parameters. This results in high empirical risk indicating that the model is not properly learning thus causing underfitting.

To overcome the above issues the deep learning models are trained in a slightly different approach with certain tricks. For instance, in a classification problem instead of using 0-1 loss, we use a proxy or surrogate loss like the log-likelihood of the correct class. We use regularization to prevent the model from being too complicated and avoid overfitting. Early stopping is another alternative to prevent overfitting. For comprehensive details we recommend [Goodfellow et al. \[2016\]](#).

2.2.1.5 Gradient based Optimizers

The optimization problem of deep neural networks is solved using a gradient-based method. The most commonly used gradient-based algorithm is called gradient descent (GD). The key idea behind GD is to start with an initial configuration of parameters θ_0 and iteratively update the parameters in the negative direction of the gradient of a function to be optimized.

Formally, consider a multi-dimensional continuous function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with parameters $\theta \in \mathbb{R}^{n \times m}$. The gradient of a function $f(\theta)$ with respect to parameters θ is defined as:

$$\nabla_{\theta} f = \begin{bmatrix} \frac{\partial f}{\partial \theta_{00}} & \cdots & \frac{\partial f}{\partial \theta_{0n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial \theta_{m0}} & \cdots & \frac{\partial f}{\partial \theta_{mn}} \end{bmatrix} \quad (2.15)$$

The iterative update of a GD is defined as:

$$\theta_{n+1} = \theta_n - \eta \nabla_{\theta} f(\theta) \quad (2.16)$$

where $\eta \in \mathbb{R}^+$ is the learning rate which controls the rate of convergence. In the case of NNs $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is a function computed by taking the average of loss function over all training samples:

$$f(\theta) = \frac{1}{M} \sum_{i=0}^M \mathcal{L}(x_i, y_i; \theta) \quad (2.17)$$

where $(x_i, y_i) \in \mathcal{D}_{\text{train}}$. For a large dataset computing the gradient-update for all training samples is computationally infeasible as it requires summation over all samples. This issue is addressed with the stochastic update rule where gradients are updated for a randomly sampled batch of samples from the data set. This makes learning tractable and also results in faster convergence. Algorithm 2.2 gives describes the pseudocode of stochastic-gradient-descent.

Algorithm 2.2 Stochastic Gradient Descent

Initialize Learning rate: η , Model parameters: θ
repeat
 Randomly sample minibatch of samples $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\} \sim p_{\text{data}}$
 Compute Mini-batch gradient $\theta_{\text{grad}} \leftarrow \frac{1}{m} \sum_{i=0}^M \nabla_{\theta} \mathcal{L}(x^{(i)}, y^{(i)}, \theta)$
 Update the parameters $\theta \leftarrow \theta - \eta \theta_{\text{grad}}$
until Convergence

In general, the optimization of deep neural networks is often difficult due to the complexity of the curvature of the loss function mainly introduced by a large number of network parameters. There are various other learning methods introduced as an improvement over SGD. Qian [1999] introduced momentum with an idea of a ball with mass moving downhill analogous to curvature of the loss function. Duchi

et al. [2011] introduced an adaptive learning rate of each parameter allowing slower update for frequently occurring features. Kingma and Ba [2014] introduced Adam an optimizer which utilizes momentum as well as the adaptive learning rate. We avoid a detailed discussion of all methods. For a detailed discussion, we refer to Goodfellow et al. [2016].

2.2.2 Generative Models

The GMs deal with the estimation of the probability distribution of data. More specifically, consider p_{data} to be a true data distribution, D_{data} be a set of training samples. The objective of generative model is to learn a distribution p_{model} using D_{data} such that p_{model} is a good approximation of p_{data} . Such an estimate provides the description of data and thus can be used to generate new samples.

Following Goodfellow [2016] we characterize GMs as: (i) explicit density models, a class of model which learn p_{model} explicitly. The common models in this class are: Restricted Boltzmann Machines (RBMs) (Salakhutdinov et al. [2007]), Sigmoid Belief Networks (SBNs) (Hinton [2009]), Deep Boltzmann Machines (DBMs) (Salakhutdinov and Hinton [2009]). One main problem in designing this class of model is the computational intractability. For details refer to (Salakhutdinov [2015], Goodfellow [2016]). (ii) implicit density models, a class of model which learns to generate samples from p_{data} without explicitly defining a density function. Some common models in this class are generative stochastic networks (GSNs) (Bengio et al. [2014]), generative adversarial networks (GANs) (Goodfellow et al. [2014]), etc.

2.2.2.1 Maximum Likelihood Principle

The above class of GMs are commonly solved using the maximum likelihood principle. Let $\mathcal{X} = \{x_1, \dots, x_N\}$ be a set of training samples, $f(x; \theta)$ be a parametric model that provides an estimate of the data distribution p_{data} using a model distribution p_{model} . Our goal is to find the configuration of parameters θ which gives the best estimate of p_{model} approximating p_{data} . Using the maximum likelihood principle the best parameters $\hat{\theta}$ is given by Equation 2.18. To further ease the optimization points are sampled from an i.i.d. distribution and the optimization is solved in the log space.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} p_{\text{model}}(\mathcal{X}; \theta) \quad (2.18)$$

$$= \underset{\theta}{\operatorname{argmax}} \prod_{i=0}^N p_{\text{model}}(x_i; \theta) \quad (2.19)$$

$$= \underset{\theta}{\operatorname{argmax}} \sum_{i=0}^N \log p_{\text{model}}(x_i; \theta) \quad (2.20)$$

Figure 2.4 gives an example of a maximum-likelihood principle.

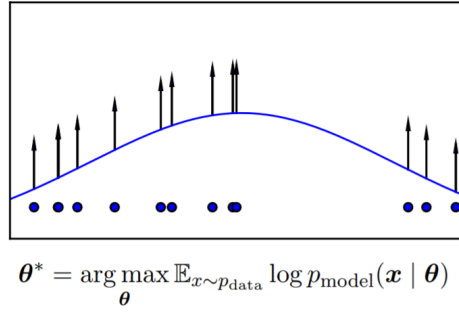


Figure 2.4: An illustration of a maximum likelihood principle shows how different data points push up different part of the density function. The maximum likelihood principle takes samples from the training set and then pushes up the probability that a model assigns to those points. Taken from [Goodfellow \[2016\]](#)

2.2.3 Generative Adversarial Networks

Learning to sample from high dimensional probability distribution p_{data} is complicated and often an intractable process. However, sampling from a simple distribution like a Gaussian is well studied and computationally efficient. The key idea behind GANs ([Goodfellow et al. \[2014\]](#)) is to take samples from simple distribution and apply a non-linear parametric transformation to get samples from the high dimensional complex distribution. This learnable transformation is optimized in an adversarial setting.

A GAN consists of two networks namely generator $\mathcal{G}$ and a discriminator $\mathcal{D}$. Consider $p(z)$ to be a prior noise distribution, $\mathcal{X}$ be a set of training samples drawn from data distribution p_{data} . The network $\mathcal{G}$ transforms samples from noise distribution $z \sim p(z)$ to samples in model distribution $\mathcal{G}(z; \theta_g)$. The network $\mathcal{D}$ takes an input sample x and estimates the probability $\mathcal{D}(x; \theta_d)$ of sample x being drawn from $\mathcal{X}$. Thus a discriminator $\mathcal{D}$ works as a binary classifier to discriminate between real samples in $\mathcal{X}$ and generated samples $\mathcal{G}(z; \theta_g)$. The objective of generator $\mathcal{G}$ is to generate samples which are similar to real samples and thus can effectively fool the discriminator at the task of classification.

2.2.4 How to train a GAN?

The objective of the generator and discriminator can be interpreted as a minimax game where the generator tries to minimize the gain of the discriminator and the discriminator tries to maximize its gain. Formally, a minimax game between $\mathcal{G}$ and $\mathcal{D}$ is expressed by Equation 2.21. The two players $\mathcal{G}$ and $\mathcal{D}$ are represented by a parametric function and are fully differentiable with respect to their input as well as parameters and thus are typically modeled by neural networks.

$$\min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{D}, \mathcal{G}) = \mathbb{E}_{x \sim p_{\text{data}}} [\log \mathcal{D}(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - \mathcal{D}(\mathcal{G}(z)))] \quad (2.21)$$

The differentiable nature of $\mathcal{G}$ and $\mathcal{D}$ allows solving the optimization using gradient-based methods. Since the input of one depends on the output of other the optimization is solved in two steps of gradient updates, first updating the parameters of θ_g and then updating θ_d . Algorithm 2.3 further illustrates the learning of $\mathcal{G}$ and $\mathcal{D}$. Figure 2.5 gives a geometric illustration of GAN training.

Algorithm 2.3 Minibatch stochastic gradient descent training of GAN (Goodfellow et al. [2014]). For each update of the generator we update the discriminator k number of times, where k is the hyperparameter

```

for number of iterations do
  for k steps do
    Sample minibatch of m noise samples from prior  $\{z^{(1)}, \dots, z^{(m)}\} \sim p_z(z)$ .
    Sample minibatch of m samples from data distribution  $\{x^{(1)}, \dots, x^{(m)}\} \sim$ 
     $p_{\text{data}}$ .
    Update the parameters of the discriminator by ascending the stochastic
    gradient on a minibatch:
       $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m [\log \mathcal{D}_{\theta_d}(x^{(i)}) + \log(1 - \mathcal{D}_{\theta_d}(z^{(i)}))]$ 
    end for
    Sample minibatch of m noise samples from prior  $\{z^{(1)}, \dots, z^{(m)}\} \sim p_z(z)$ .
    Update the generator by descending the stochastic gradient on a minibatch:
       $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m [\log(1 - \mathcal{D}_{\theta_d}(\mathcal{G}_{\theta_g}(z^{(i)})))]$ 
  end for

```

At each step of training, the cost of one player depends on the decision of the other one. However, both players have no control over the strategy of each other. This results in a zero-sum game where the loss of one player is equivalent to the benefit of the other one. If both players are optimal then there exists an equilibrium between them referred to as Nash Equilibrium. In the NN setting of a GAN, the optimization problem of both $\mathcal{G}$ and $\mathcal{D}$ is non-convex and the only plausible solution is a local minimum which implies an existence of a local Nash Equilibrium a state in which cost of $\mathcal{G}$ is minimal w.r.t parameters θ_g and of $\mathcal{D}$ is minimal w.r.t. parameters θ_d (Ratliff et al. [2013]).

2.2.5 GAN: An Information Theoretic Perspective

In this section, we will study the convergence properties of Algorithm 2.3. Consider p_{data} to be the probability distribution of the data and p_g to be the probability distribution learned by the generator $\mathcal{G}$. We want to measure how well the probability distribution p_g approximates the distribution p_{data} . To study the convergence we will assume that the $\mathcal{G}$ and $\mathcal{D}$ networks are of infinite capacity. We will first introduce Kullback-Leibler (KL) Divergence and Jensen Shannon Divergence and will then continue our discussion of optimality.

2.2.5.1 Kullback-Leibler Divergence

The KL divergence (Kullback and Leibler [1951]) is a measure of discrepancy between two probability distributions. From an information theory point of view, a KL-divergence between two probability distributions P and Q is a measure of an extra number of bits required to code samples from P when using a code of samples from Q . It tells us how much information is lost when Q is used to approximate P . Formally, it is defined as:

$$D_{KL}(P||Q) = \mathbb{E}_{x \sim P} \log \frac{P}{Q} \quad (2.22)$$

The KL divergence is a non-negative quantity and is equal to zero if and only if $P = Q$. It is not a proper metric as it is asymmetric and does not obey the triangle inequality.

2.2.5.2 Jensen-Shannon Divergence

Jensen-Shannon divergence (JSD) is another common measure of similarity between two probability distributions. It is symmetric and a smoothed variant of the KL divergence and is defined as:

$$JSD(P, Q) = KL(P||M) + KL(Q||M) \quad (2.23)$$

where $M = \frac{P+Q}{2}$. The choice of M as the average of P and Q further gives an upper bound on the divergence. In an information theory sense, JSD is regarded as the mutual information between a mixture distribution of P and Q and a random variable indicating the switch between P and Q to produce the mixture.

2.2.5.3 Optimality of minimax

The optimal solution to $V(G, D)$ corresponds to finding a parameter configuration θ_g that minimizes the objective and θ_d that maximizes the objective. The optimal discriminator $\mathcal{D}$ for a fixed $\mathcal{G}$ is given by:

$$D_G^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)} \quad (2.24)$$

For an optimal discriminator the optimization function can be formulated as:

$$C(G) = \max_D V(G, D) \quad (2.25)$$

$$= \mathbb{E}_{x \sim p_{\text{data}}} [\log \mathcal{D}^*(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - \mathcal{D}^*(\mathcal{G}(z)))] \quad (2.26)$$

$$= \mathbb{E}_{x \sim p_{\text{data}}} \left[\log \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)} \right] + \mathbb{E}_{z \sim p(z)} \left[\log \frac{p_g(x)}{p_{\text{data}}(x) + p_g(x)} \right] \quad (2.27)$$

For $p_g = p_{\text{data}}$ the loss function $C(G)$ can further be expressed as:

$$C(G) = -\log 4 + 2 \cdot JSD(p_{\text{data}}||p_g) \quad (2.28)$$

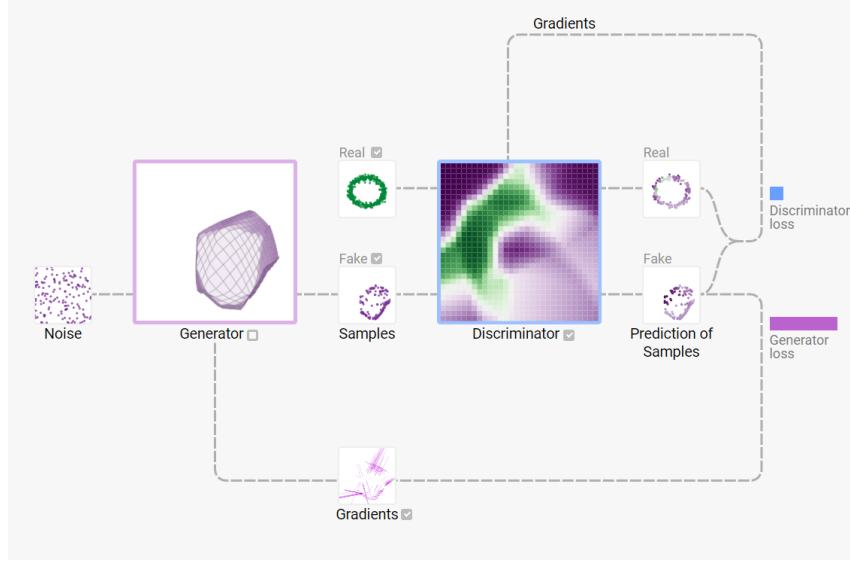


Figure 2.5: A geometric illustration of GAN training. Generator learns to transform samples from a simple noise distribution $\mathcal{N}(0, 1)$ to complex data distribution. Taken from Kahng et al. [2018]

As discussed in Section 2.2.5.2 the JSD is non-negative and takes a minimum when the two distributions are equal. The optimization problem has a global optimum when $p_g = p_{\text{data}}$ i.e. with an infinite capacity the generative network can perfectly model the data distribution. Thus, in the training process, the optimal discriminator aims at estimating the ratio of probability densities as given in Equation 2.24 and the optimal generator aims at minimizing the JSD between p_g and p_{data} .

2.2.5.4 Challenges in GAN training

The training of GANs requires finding an equilibrium between generator and discriminator. At each step, $\mathcal{G}$ and $\mathcal{D}$ try to undo the progress of each other which makes it difficult to monitor their convergence. Thus, the training of GANs is often associated with instability. The two main problems are: (i) Oscillatory behavior across modes: This is a situation where the generator progressively switches from generating one kind of sample to another without reaching an equilibrium condition. (ii) Mode collapse: In mode collapse the generator starts mapping multiple latent noise representations to a single point in the data distribution. The more commonly encountered form of a model collapse in training of GANs is known as partial mode collapse which occurs when the generator maps only to a smaller set of training examples resulting in a low diversity of generated samples. This is further explained by the asymmetric nature of minimax that is to say:

$$\min_G \max_D V(G, D) \neq \max_D \min_G V(G, D) \quad (2.29)$$

In the minimax situation the optimal generator draws sample from the data distribution.

$$G^* = \min_G \max_D V(G, D) \quad (2.30)$$

In the case of maxmin, the optimization is first for generator and then discriminator. As a result, it maps every noise distribution to a single data sample. The update methods do not differentiate between minimax and maxmin. As a result in some cases the optimization turns out to solve the maxmin resulting in the mode collapse. The severity of mode collapse also depends on the diversity of training data. It is more common when training data consists of samples of only a few modes.

2.2.6 Evaluation of GANs

Over the past few years, various improvements have been proposed for GAN training (e.g. Karras et al. [2017], Arjovsky et al. [2017], Petzka et al. [2017]). Despite all the improvements, the evaluation and comparison of models is still a difficult task. In this work, we use two widely accepted measures namely the inception score (Salimans et al. [2016]) and the Fréchet inception distance (Heusel et al. [2017]).

To evaluate the quality of the generated samples we compute Inception Score (IS) and Fréchet Inception Distance (FID). In the following section, we will look into the formalism of IS and FID. For the evaluation of generated samples, we use a held out set of 40 speakers.

2.2.6.1 Inception Score

The IS (Salimans et al. [2016]) was introduced as a measure to capture the diversity of generated samples. It was originally proposed for the evaluation of GANs trained to generate images. The two key ideas behind the IS are: (i) an image x which contains a meaningful object has a low entropy of the label distribution $p(y|x)$ and (ii) for the high diversity in generated samples the marginal distribution $\int p(y|x = G(z))dz$ should have a high entropy. By combining the two ideas IS is defined as:

$$\text{IS} = \exp^{\mathbb{E}_{x \sim p_g}[KL(p(y|x)||p(y))]} \quad (2.31)$$

where p_g refers to the generated distribution and exp is done for scaling the scores to comparable range. For m samples and K classes the term $\mathbb{E}_x[KL(p(y|x)||p(y))]$ is defined as:

$$\mathbb{E}_x[KL(p(y|x)||p(y))] = \frac{1}{m} \sum_{i=0}^{m-1} \sum_{k=0}^{K-1} p(y_k|X_i) \log\left(\frac{p(y_k|X_i)}{p(y_k)}\right) \quad (2.32)$$

The intuition behind the IS also follows for speech samples. If the generated sample contains utterance properties of a unique speaker the conditional label distribution should have a low entropy. Likewise, for the diversity in speakers in generated samples, the marginal $\int p(y|x = G(z))dz$ should have a high entropy. For the IS we use our speaker detection network discussed in Section 4.1.1. The higher

the value of the IS the better is the quality of generated samples. The minimum value of the IS is zero implying the generated samples are an average distribution of all modes present in the data and the model has failed to generate meaningful samples. The IS is bounded from above by support size m . For details, we refer to Heusel et al. [2017].

2.2.6.2 Fréchet Inception Distance

The disadvantage of the IS is that it only uses the generated samples and does not perform a comparison with the real-world samples. To address this problem Heusel et al. [2017] introduced the FID, a distance measure to compare the statistics of real and generated data. Since the Gaussian is the maximum entropy distribution, FID assumes the feature encoding follows multivariate Gaussians. Consider $(\mu_{\text{real}}, C_{\text{real}})$ be statistics of the real data $\mathcal{X}_{\text{real}}$ and $(\mu_{\text{gen}}, C_{\text{gen}})$ be statistics of the generated data $\mathcal{X}_{\text{gen}}$. The FID between two multivariate Gaussians is defined as:

$$\text{FID} = \|\mu_{\text{real}} - \mu_{\text{gen}}\|_2^2 + \text{Tr}(C_{\text{real}} + C_{\text{gen}} - (2C_{\text{real}}C_{\text{gen}})^{1/2}) \quad (2.33)$$

where $\text{Tr}(X)$ denotes trace of the matrix X . A lower value of the FID means that the distribution of the generated samples is close to the distribution of the real samples. Thus, the lower the score the better is the model.

2.2.7 Variational Auto-encoder

The Variational Autoencoder (VAE) introduced by Kingma and Welling [2013] is an implicit density generative model that learns to encode data. One problem in learning a parametric function to transform a simple distribution to a complex distribution is that for high dimensional distributions it requires sampling of a large number of noise samples $z = \{z_1, \dots, z_N\}$ to estimate the probability distribution of the data $p_{\text{data}} \approx \frac{1}{N} \sum_i p_g(X|z_i)$. A large value of N often results in intractability and other training issues. The main idea behind the VAE is to give generator those latent samples z that are more likely to produce samples from p_{data} . This is achieved by conditioning the distribution of latent z on the training samples X .

Formally, consider a parametric inference network that outputs statistics of $q(z|X)$, a generator network that outputs a distribution $p_g(X|z)$. A VAE learns to approximate the posteriors $p_g(z|X)$ with $q(z|X)$. This is achieved by optimizing a variational lower bound

$$V(x; \theta_1, \theta_2) = -D_{KL}(q_{\theta_1}(z|x)) \| p_{\theta_2}(z|x) + \mathbb{E}_{q(z|x)} [\log p_{\theta_2}(x|z)] \quad (2.34)$$

where θ_1 and θ_2 are parameters of the inference and generator networks. The gradients of the lower bound with respect to parameters θ_1 are intractable. A VAE uses the variational calculus to cast the inference as an optimization problem. Consider a given intractable posterior probability distribution $p_{\theta_1}(z|x)$, using the variational principle we solve an optimization problem over a family of tractable

distributions $\mathcal{F}$. For $q_{\theta_2}(z|x) \in \mathcal{F}$ the optimal approximation is given by:

$$q^* = \underset{\theta_1}{\operatorname{argmin}} D_{KL}(q_{\theta_1}(z|x) || p_{\theta_2}(z|x)) \quad (2.35)$$

We refer to [Doersch \[2016\]](#) for a more comprehensive discussion of VAEs.

2.2.8 CycleGAN

The cycleGAN ([Zhu et al. \[2017\]](#)) is a generative model proposed for unsupervised translation of images between two domains. It is specifically useful for applications like domain translation, style transfer, etc. The key idea behind cycleGAN is the cycle consistency loss. Consider two sets of images of two domains $X = \{x_1, \dots, x_N\}$ and $Y = \{y_1, \dots, y_M\}$. The cycleGAN consists of a generator and discriminator for each domain. Let F and G be two generator networks defined as:

$$F : X \rightarrow Y \quad (2.36)$$

$$G : Y \rightarrow X \quad (2.37)$$

Let D_X and D_Y be two discriminator networks defined as:

$$D_X : X \rightarrow \mathbb{R} \quad (2.38)$$

$$D_Y : Y \rightarrow \mathbb{R} \quad (2.39)$$

Discriminator D_X is trained to discriminate between original images in X and generated images $G(X)$ likewise D_Y is trained to discriminate between original images in Y and generated images $F(Y)$. The combined training loss of two GANs is expressed as:

$$V_{\text{GAN}}(F, D_Y, G, D_X) = V_{\text{GAN}}(F, D_Y, X, Y) + V_{\text{GAN}}(G, D_X, X, Y) \quad (2.40)$$

One essential constraint imposed in cycleGAN is the cycle consistency loss. It ensures that certain attributes of the original domain are preserved when an image is translated. Specifically, this is done by constraining F and G to be the inverse mapping of each other. Specifically:

$$F \circ G = I_Y \quad (2.41)$$

$$G \circ F = I_X \quad (2.42)$$

The cyclic loss term is defined as:

$$V_{\text{cyc}}(F, D_Y, G, D_X) = \|X - G(F(X))\|_1 + \|Y - F(G(X))\|_1 \quad (2.43)$$

The full optimization term is expressed as:

$$F^*, G^* = \underset{F, G}{\operatorname{argmin}} \max_{D_X, D_Y} V_{\text{GAN}}(F, G, D_X, D_Y) + V_{\text{cyc}}(F, D_Y, G, D_X) \quad (2.44)$$

Figure 2.6 gives a geometric view of cycleGAN. The cycle consistency loss prevents the problem of mode collapse by forcing the network to retain certain features of the original domain.

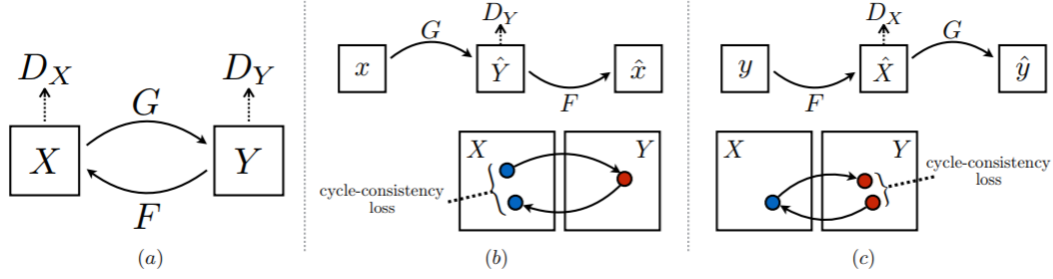


Figure 2.6: An illustration of mapping in cycleGAN. (a) The model contains a pair of generator discriminator networks (F, D_Y) and (G, D_X) where $F : X \rightarrow Y$ and $G : Y \rightarrow X$ are generator networks and $D_Y : Y \rightarrow \mathbb{R}$ and $D_X : X \rightarrow \mathbb{R}$ are corresponding discriminator networks, (b) and (c) demonstrate cycle consistency for two domains. Taken from [Zhu et al. \[2017\]](#)

2.2.9 Unsupervised Image-to-Image Translation

Unsupervised image-to-image translation (UNIT) is another GAN-based framework for the unpaired translation of images between two domain. In this thesis, our work is inspired by the UNIT framework. The UNIT ([Liu et al. \[2017\]](#)) is based on the assumption that two domains X and Y have an underlying joint probability distribution $P_{X,Y}(x, y)$ and the objective is to approximate $P_{X,Y}(x, y)$ from two marginal distributions $P_X(x)$ and $P_Y(y)$. Inferring a joint distribution from marginal distribution is an ill-posed problem. The UNIT approximates the joint probability distribution under the assumption that the two domains have a shared latent space $\mathcal{Z}$.

Unlike cycleGAN in UNIT the two generators are expressed as encoder-decoder pairs (E_1, G_1) for domain X , (E_2, G_2) for domain Y and the corresponding discriminators D_1 and D_2 . For any pair of images (x, y) where $x \in X$ and $y \in Y$ there exist a representation in shared latent space $z \in \mathcal{Z}$ as

$$z = E_1(x) = E_2(y) \quad (2.45)$$

and the mapping from $z \in \mathcal{Z}$ to image space is given by:

$$x = G_1(z) \quad (2.46)$$

$$y = G_2(z) \quad (2.47)$$

The concept of shared latent space is implemented by imposing the following two conditions: (i) Weight sharing of the last layer of the encoder network and the first layer of the decoder network, (ii) cycle consistency loss. Condition (i) is a more strict condition since it also implies cycle consistency. Condition (ii) is imposed as a constraint to restrict the search space of a network and to avoid problems like the mode collapse. Figure 2.7 gives a geometric view of shared latent space and the UNIT network.

The VAE-GAN loss of two generators is defined as:

$$V_{\text{VAE-GAN}}(E_1, G_1, E_2, G_2, D_1, D_2) = V_{\text{VAE}}(E_1, G_1, X, Y) + V_{\text{GAN}}(E_2, G_1, D_1, X, Y) \\ + V_{\text{VAE}}(E_2, G_2, X, Y) + V_{\text{GAN}}(E_1, G_2, D_2, X, Y) \quad (2.48)$$

The cycle consistency loss is expressed like a VAE objective function given by:

$$V_{\text{cyc}_1}(E_1, G_1, E_2, G_2) = \lambda_1 D_{KL}(q_1(z_x|x_X)||p_\eta(z)) + \lambda_2 D_{KL}(q_2(z_y|x_X^{X2Y})||p_\eta(z)) \\ + \lambda_3 \mathbb{E}_{z_y \sim q_2(z_y|x_X^{X2Y})} [\log p_{G_1}(x_X|z_y)] \quad (2.49)$$

$$V_{\text{cyc}_2}(E_2, G_2, E_1, G_1) = \lambda_1 D_{KL}(q_2(z_2|y_Y)||p_\eta(z)) + \lambda_2 D_{KL}(q_1(z_x|y_Y^{Y2X})||p_\eta(z)) \\ + \lambda_3 \mathbb{E}_{z_x \sim q_1(z_x|y_Y^{Y2X})} [\log p_{G_1}(x_X|z_x)] \quad (2.50)$$

$$V_{\text{cyc}}(E_1, G_1, E_2, G_2) = V_{\text{cyc}_1}(E_1, G_1, E_2, G_2) + V_{\text{cyc}_2}(E_2, G_2, E_1, G_1) \quad (2.51)$$

where $p_\eta(z) = \mathcal{N}(z|0, I)$ is a prior distribution assumed to be Gaussian, $z_x = E_1(x) + \eta$, $z_y = E_2(y) + \eta$ and $\eta \sim p_\eta(z)$. The KL term ensures that the latent space distribution is close to the prior and the negative log-likelihood term is similar to the consistency term in the cycleGAN and ensures $G_1(E_2(G_2(E_1(X)))) \sim X$ and $G_1(E_2(G_2(E_1(Y)))) \sim Y$.

The full optimization problem of the UNIT is expressed as:

$$E_1^*, E_2^*, G_1^*, G_2^*, D_1^*, D_2^* = \underset{E_1, G_1, E_2, G_2}{\operatorname{argmin}} \max_{D_1, D_2} V_{\text{VAE-GAN}}(E_1, G_1, E_2, G_2, D_1, D_2) \\ + V_{\text{cyc}}(E_1, G_1, E_2, G_2) \quad (2.52)$$

2.3 Deep Generative Models for Audio Signals

In this section, we review recent work on deep generative models for audio signals. The recent developments in generative models in particular GANs have already achieved a lot of success for image data. However, the application to audio signals is still in the early stage. The WaveNet (Oord et al. [2016]) proposed an autoregressive generative model for high-quality audio synthesis. One key drawback of this approach is that it works in the time domain and its autoregressive nature results in slow inference for audio data sampled at high frequency.

One alternative to this approach is to use a compressed STFT representation instead of the time-domain representation of a signal. In our previous discussion, we looked at the problem of consistency and phase reconstruction associated with this approach. In a recent work Engel et al. [2019] avoid the phase reconstruction and proposed a generative model that utilizes the full time-frequency representation and at inference time uses the ISTFT operation to get the time-domain waveform.

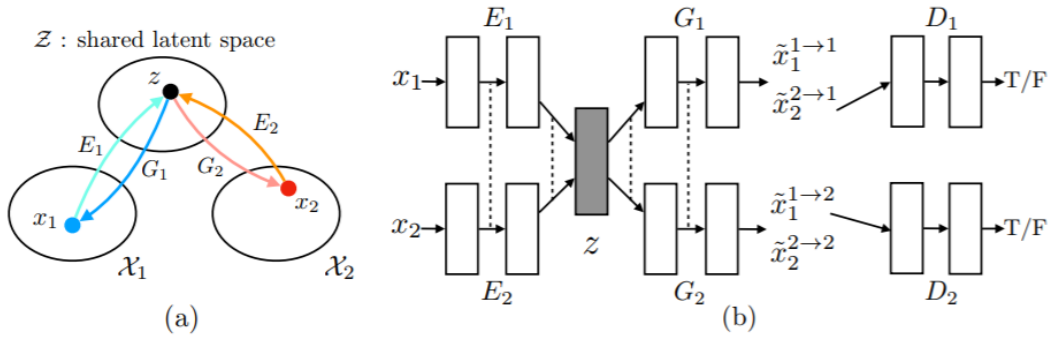


Figure 2.7: UNIT Network: (E_1, G_1) and (E_2, G_2) are two pairs of VAEs and D_1 and D_2 are two discriminators corresponding to domain X and Y , respectively. (a) An Illustration of shared latent space $\mathcal{Z}$. The UNIT assumes the pair (x_1, x_2) can be represented by a shared latent embedding $z \in \mathcal{Z}$. (b) An example of UNIT network where dotted line illustrates weight sharing. Taken from Liu et al. [2017].

To deal with the convoluted nature of phase spectra the authors transform it to an alternative representation namely the instantaneous frequency (IF). In Marafioti et al. [2019] the authors proposed a GAN network with an additional constraint to generate a consistent spectrogram representation in an adversarial setting. The consistency condition speeds up the learning of an iterative algorithm for phase reconstruction thus provides a better quality of the audio synthesis.

The domain adaptation of audio signals in particular of speech data offers a promising direction to improve automatic speech recognition (ASR) systems, for the development of robust speaker detection systems, privacy in end-to-end dialogue systems and many more exciting problems. Hosseini-Asl et al. [2018] proposed a cycleGAN-based network using the spectrogram representation for cross-domain adaptation of speech signals. The approach makes use of multiple discriminators based on a frequency band to allow the generator to focus on fine-grained features.

In Kaneko and Kameoka [2017] the authors proposed an architecture utilizing the cycleGAN framework for converting the voice of one speaker to another. The approach is evaluated on the VCC 2016 dataset (Nakashika et al. [2016]) of 10 speakers. In Huang et al. [2018] the authors proposed a VAE-based architecture for voice conversion. The network jointly uses two types of spectral features in the VAE setting. In another recent work Qian et al. [2019], the authors proposed an autoencoder framework for zero-shot voice conversion. The conversion is done by incorporating speaker embedding obtained from a separate pretrained speaker classification network. The pretrained network is trained on the task of speaker identification on a large corpus of 3 549 speakers. Later for zero-shot voice conversion the decoder of an autoencoder network is conditioned on the embeddings obtained from the pretrained model.

Chapter 3

Cross-domain Speech-to-Speech Synthesis

In the previous chapter, we discussed the recent developments of GANs and their applications to the domain translation for images. Following our discussion in Chapter 1, in this chapter we propose our work on unsupervised cross-domain speech-to-speech synthesis. We will start our discussion with an introduction to the problem. We will then formulate it specifically in the setting of the unsupervised image-to-image translation network previously discussed in Section 2.2.9.

We use the time-frequency representation of speech signals to train the network. We use the STFT transformation to obtain the time-frequency representation. In Chapter 1 we discussed the problem of time-domain waveform reconstruction from the modified or generated time-frequency representation. In this chapter, we present our solution to the problem. We utilize the consistency condition of the time-frequency representation to constrain the optimization problem of the GAN. The time-frequency representation comprises of two parts namely the magnitude and phase spectrogram. We propose two experimental setups. In the first one, we use the consistency condition of the magnitude-spectrogram for training the network and at inference time we use the Griffin-Lim algorithm (GLA) to reconstruct the time-domain speech waveform. In the second one, we use the full time-frequency representation for training the network. Furthermore, at inference time we straight-away perform the ISTFT operation on the time-frequency representation to obtain the time-domain waveform and thus avoid the computational cost incurred by the GLA.

3.1 Speech-to-Speech Synthesis

In the context of our work, we formulate domain translation as the task to translate the voice of male speakers to female speakers and vice versa. In Section 3.1.1 we formulate it as a learning problem. In Section 3.1.1 we discuss the details of our implementation. Finally, in Section 3.2.3, we discuss the full architecture of the network.

3.1.1 Problem definition

Consider $\mathcal{X}_1$ and $\mathcal{X}_2$ be two domains representing **male** and **female** speakers. For a speech sample $x_1 \in \mathcal{X}$ our objective is to find the corresponding sample $x_2 \in \mathcal{X}_2$. In our work, we achieve this under the shared space assumption. Specifically, for any pair (x_1, x_2) we assume there exists a $z \in \mathcal{Z}$ which can represent a pair of samples across two domains. In our task $\mathcal{Z}$ is interpreted as a shared space of a voice sample of a male and female speaker. This should encode domain independent information. For instance, linguistic features of the paired voice samples and the phonetic variations of speakers.

To obtain a shared space $\mathcal{Z}$, we define parametric functions E_1, E_2, G_1, G_2 such that, for a corresponding pair (x_1, x_2) we assume there exist a latent representation z such that $x_1 = G_1(z)$ and $x_2 = G_2(z)$. To implement this E_1 and E_2 are modeled as multi-layered residual networks where the corresponding last layers share parameters. Specifically, this is done by decomposing two encoder networks E_1 and E_2 as $E_1 = E_h \circ E_{\bar{1}}$ and $E_2 = E_h \circ E_{\bar{2}}$, where $E_{\bar{1}}, E_{\bar{2}}$ are domain specific encoders and E_h is a shared parametric transformation mapping to the latent space $\mathcal{Z}$. The shared space assumption already implies cycle consistency as discussed in Section 2.2.8. To further regularize and constrain the model we define the cycle consistency loss as explained in Section 2.2.9.

At the time of inference, for any $x_1 \in \mathcal{X}_1$ or $x_2 \in \mathcal{X}_2$ we can obtain the corresponding cross domain translation by the compositional mapping $x_{1 \rightarrow 2} = G_2(E_1(x_1))$ or $x_{2 \rightarrow 1} = G_1(E_2(x_2))$. We implement parametric functions as a VAE-GAN. Specifically, the pairs (E_1, G_1) and (E_2, G_2) are implemented as VAE generator networks. To obtain a better quality of the cross domain translation the VAEs are optimized in an adversarial setup. Specifically, we define two discriminators D_1 and D_2 where D_1 discriminates between fake female-to-male samples and original male samples. Likewise, D_2 discriminates between fake male-to-female samples and original female samples. The full optimization problem is expressed as:

$$\begin{aligned} E_1^*, E_2^*, G_1^*, G_2^*, D_1^*, D_2^* = & \underset{E_1, G_1, E_2, G_2}{\operatorname{argmin}} \max_{D_1, D_2} V_{\text{VAE}}(E_1, G_1, \mathcal{X}_1, \mathcal{X}_2) + V_{\text{GAN}}(E_2, G_1, D_1, \mathcal{X}_1, \mathcal{X}_2) \\ & + V_{\text{VAE}}(E_2, G_2, \mathcal{X}_1, \mathcal{X}_2) + V_{\text{GAN}}(E_1, G_2, D_2, \mathcal{X}_1, \mathcal{X}_2) \\ & + V_{\text{cyc}}(E_1, G_1, E_2, G_2, \mathcal{X}_1, \mathcal{X}_2) \end{aligned} \quad (3.1)$$

3.1.2 Magnitude Spectrogram Generation

As discussed in Section 2.1.2, we work with four different scales of the magnitude spectrogram: magnitude, log magnitude, mel, and log mel. The log scale enhances the lower frequency components related to the perceptual quality of the waveform to human ears. Likewise, the mel scale transforms the real frequency to the perceived frequency enhancing components relating to the perceptual nature of sound. For details, we refer to our discussion in Section 2.1.2.

3.1.2.1 Spectrogram Consistency Loss

In Section 2.1.3 we looked at the problem associated with the usage of the magnitude spectrogram for training neural networks. The chain of transformation applied by a neural network on a magnitude spectrogram does not preserve the overlap factor used in the STFT operation. Since the STFT provides an overcomplete representation, to reconstruct a coherent waveform from a modified magnitude spectrogram representation requires the right combination of phase and magnitude at each frequency bin. As a consequence, the waveform reconstruction is considered as an ill-posed problem.

The GLA discussed in Section 2.1.3.2 is a popular technique used for estimating the closest waveform from the magnitude spectrogram. The GLA estimates the unknown phase by iteratively performing ISTFT and STFT operation to project the spectrogram between the time and frequency domain and thus minimizing the projection error. A critical requirement for the convergence of the GLA is the consistency of the spectrogram representation, that is if there is a valid time-domain waveform associated with the spectrogram. For a spectrogram of the original signal, this consistency is guaranteed by the sufficient overlap factor in the STFT transformation. However, for a generated spectrogram, it is not clear if there is any corresponding time-domain waveform. For details we refer to our discussion in Section 2.1.3.1.

In Auger et al. [2012] the authors proposed a relationship between the phase and magnitude part of the spectrogram. Furthermore, they exploit the relationship and derive a necessary condition for the consistency of any log magnitude matrix $\log(\hat{M}_{\hat{g}}(x, \omega))$, which for a Gaussian window under the STFT operation is given by following equation:

$$(\lambda \frac{\partial^2}{\partial x^2} + \lambda^{-1} \frac{\partial^2}{\partial \omega^2}) \log(\hat{M}_{\hat{g}}(x, \omega)) = -2\pi \quad (3.2)$$

Equation 3.2 resembles the condition of analytic functions where the Laplacian of the log magnitude matrix $\log(\hat{M}_{\hat{g}}(x, \omega))$ is constant. Thus, the consistency of the spectrogram could be interpreted as finding the analytic space of spectrograms. For further details we recommend Auger et al. [2012].

The discrete version of Equation 3.2 for the log magnitude matrix $\log(\hat{M}_{\hat{g}}[m, n])$ under the Gaussian window g is approximated as

$$(\frac{\lambda}{a^2} \frac{\partial^2}{\partial n^2} + K^2 \lambda^{-1} \frac{\partial^2}{\partial m^2}) \log(\hat{M}_{\hat{g}}[m, n]) \approx -2\pi \quad (3.3)$$

where a is the stride length, K is the number of frequency bins, n is the time axis and m is the frequency axis in the STFT matrix.

Marafioti et al. [2019] proposed a loss term for the adversarial generation of the spectrogram exploiting the consistency condition in Equation 3.3. Equation 3.3 is prone to the approximation error which is proportional to the overlap in the STFT operation. Marafioti et al. [2019] define a variant of a measure inspired by the Pearson correlation and further support it by extensive empirical results. To define

a measure formally, consider $\tilde{M}$ to be a generated magnitude spectrogram. Then the consistency $\rho(\tilde{M})$ of $\tilde{M}$ is given as

$$\rho(\tilde{M}) = r(\text{DM}_n, \text{DM}_m), \quad (3.4)$$

$$\text{DM}_n = \left| \partial_n^2 \tilde{M} + \frac{\pi a^2}{\lambda} \right|, \quad (3.5)$$

$$\text{DM}_m = \left| \partial_m^2 \tilde{M} + \frac{\pi \lambda}{K^2} \right|, \quad (3.6)$$

where $r(x, y)$ is the sample Pearson correlation coefficient of a paired set of samples (x, y) . If the spectrogram is consistent Equation 3.3 holds and $\rho(\tilde{M}) \approx 1$. Likewise, $\rho(\tilde{M}) \approx 0$ for an inconsistent spectrogram. Extending this idea to GANs the consistency loss for adversarial training is defined as

$$\gamma = \left| \mathbb{E}_{\hat{M} \sim p_{\hat{M}_{\text{real}}}} [\rho(\hat{M})] - \mathbb{E}_{\hat{M} \sim p_{\hat{M}_{\text{fake}}}} [\rho(\hat{M})] \right| \quad (3.7)$$

where $p_{\hat{M}_{\text{real}}}$ represents the probability distribution of real spectrograms and $p_{\hat{M}_{\text{fake}}}$ represents the probability distribution of fake spectrograms. As γ approaches zero the consistency term $\rho(\tilde{M})$ will approach 1. Thus, adding γ as a constraint to the GAN loss will restrict the network to generate consistent spectrograms. For further details we refer to Marafioti et al. [2019].

Following the above formulation, we define the spectrogram consistency loss in our speech-to-speech synthesis framework. For a source spectrogram $x_1 \in \mathcal{X}_1$ we define the consistency of the cross-domain spectrogram $x_2 = G_2(E_1(x_1))$ as γ_{x_2} . Conversely, we also define γ_{x_1} and the final consistency term is given by γ as:

$$\gamma_{x_2} = \left| \mathbb{E}_{x_2 \sim p_{\mathcal{X}_2}} [\rho(x_2)] - \mathbb{E}_{x_1 \sim p_{\mathcal{X}_1}} [\rho(G_2(E_1(x_1)))] \right|, \quad (3.8)$$

$$\gamma_{x_1} = \left| \mathbb{E}_{x_1 \sim p_{\mathcal{X}_1}} [\rho(x_1)] - \mathbb{E}_{x_2 \sim p_{\mathcal{X}_2}} [\rho(G_1(E_2(x_2)))] \right|, \quad (3.9)$$

$$\gamma = \gamma_{x_1} + \gamma_{x_2} \quad (3.10)$$

We add $\lambda \cdot \gamma$ to our optimization problem given in Equation 3.1, where λ is a hyperparameter to control the contribution of the consistency loss. Since the consistency relation in Equation 3.3 only holds for the magnitude and log magnitude spectrogram we omit the experiments for the mel scale representation.

3.1.3 Full Spectrogram Generation

In the second setup, we trained our network utilizing the full spectrogram representation. As a consequence, at inference time we avoid the use of an iterative algorithm and straightaway use the ISTFT operation to reconstruct the time domain speech waveform. This further helps to avoid additional computational costs and results in a robust inference. Since the jump discontinuities in the phase spectrogram make it difficult to interpret the perceptual nature of sound we transform

the phase spectrogram into the instantaneous frequency (IF) representation. The IF is obtained by first unwrapping the phase along the time axis and then taking the finite difference along the frequency axis. The unwrapping of the phase is helpful to avoid the jump discontinuities which makes the IF a plausible choice over the phase spectrogram. For the full spectrogram, we limit our discussion to the magnitude and log magnitude scale. This is mainly due to the lack of an interpretation of the mel representation of the IF.

3.1.3.1 Spectrogram Consistency Loss

The full spectrogram is a matrix of complex numbers. We further express it in polar coordinates which is composed of two parts magnitude and phase representation. As previously discussed in Section 2.1.3.1, for any spectrogram W to be consistent it should be in a null space of the operator $\mathcal{F}$, that is to say $\mathcal{F}(W) = 0$. Since in this experiment, we reconstruct the full spectrogram instead of using the phase-magnitude relationship discussed in Section 3.1.2.1 here we directly exploit the projection error introduced in Section 2.1.3.1.

Consider $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$ be two unpaired full spectrograms in two domains. Let $\hat{M}_{x_1}$ and $\hat{P}_{x_1}$ be the magnitude and IF part of a sample x_1 likewise $\hat{M}_{x_2}$ and $\hat{P}_{x_2}$ be the magnitude and IF part of a sample x_2 . The reconstruction consistency loss for two generators is defined as γ_{gen}

$$\hat{M}_{\tilde{x}_1}, \hat{P}_{\tilde{x}_1} = G_1 \circ E_1(\hat{M}_{x_1}, \hat{P}_{x_1}), \quad \hat{M}_{\tilde{x}_2}, \hat{P}_{\tilde{x}_2} = G_2 \circ E_2(\hat{M}_{x_2}, \hat{P}_{x_2}), \quad (3.11)$$

$$\tilde{M}_{x_1}, \tilde{P}_{x_1} = \text{STFT}(\text{ISTFT}(\hat{M}_{x_1}, \hat{P}_{x_1})), \quad \tilde{M}_{x_2}, \tilde{P}_{x_2} = \text{STFT}(\text{ISTFT}(\hat{M}_{x_2}, \hat{P}_{x_2})), \quad (3.12)$$

$$\gamma_{x_1} = w_1 \|\tilde{M}_{x_1} - \hat{M}_{x_1}\|_1 + (1 - w_1) \|\tilde{P}_{x_1} - \hat{P}_{x_1}\|_1, \quad (3.13)$$

$$\gamma_{x_2} = w_2 \|\tilde{M}_{x_2} - \hat{M}_{x_2}\|_1 + (1 - w_2) \|\tilde{P}_{x_2} - \hat{P}_{x_2}\|_1, \quad (3.14)$$

$$\gamma_{\text{gen}} = \gamma_{x_1} + \gamma_{x_2} \quad (3.15)$$

where w_1 and w_2 are additional hyperparameters controlling the contribution of the magnitude and phase loss. Equation 3.12 is essentially a single iteration of the GLA applied to the generated spectrogram. Since we do not have any ground truth spectrograms for the cross-domain translation, to ensure consistency we define a cyclic loss term γ_{cyc} as follows:

$$\hat{M}_{\tilde{x}_1}, \hat{P}_{\tilde{x}_1} = G_1 \circ E_2 \circ G_2 \circ E_1(\hat{M}_{x_1}, \hat{P}_{x_1}), \quad \hat{M}_{\tilde{x}_2}, \hat{P}_{\tilde{x}_2} = G_2 \circ E_1 \circ G_1 \circ E_2(\hat{M}_{x_2}, \hat{P}_{x_2}), \quad (3.16)$$

$$\tilde{M}_{x_1}, \tilde{P}_{x_1} = \text{STFT}(\text{ISTFT}(\hat{M}_{x_1}, \hat{P}_{\tilde{x}_1})), \quad \tilde{M}_{x_2}, \tilde{P}_{x_2} = \text{STFT}(\text{ISTFT}(\hat{M}_{x_2}, \hat{P}_{\tilde{x}_2})), \quad (3.17)$$

$$\gamma_{x_1} = w_1 \|\tilde{M}_{x_1} - \hat{M}_{x_1}\|_1 + (1 - w_1) \|\tilde{P}_{x_1} - \hat{P}_{x_1}\|_1, \quad (3.18)$$

$$\gamma_{x_2} = w_2 \|\tilde{M}_{x_2} - \hat{M}_{x_2}\|_1 + (1 - w_2) \|\tilde{P}_{x_2} - \hat{P}_{x_2}\|_1, \quad (3.19)$$

$$\gamma_{\text{cyc}} = \gamma_{x_1} + \gamma_{x_2} \quad (3.20)$$

The full consistency loss γ is defined as

$$\gamma = \gamma_{\text{recon}} + \gamma_{\text{cyc}} \quad (3.21)$$

Similar to the previous case we constrain the optimization problem given in Equation 3.1 with an additional loss term $\lambda \cdot \gamma$ where λ is a hyperparameter to control the contribution of consistency loss.

In our experiments in Section 4.2, we observed the quality of magnitude spectrogram starts degrading after 10,000 iterations. We attribute this to the sparse nature of the magnitude spectrogram compared to the dense nature of the IF. We hypothesize that due to this the loss contribution of the IF starts dominating the contribution of the magnitude thus degrading the quality of the magnitude. To address this problem, we consider the weighted combination of the magnitude and phase loss term in Equation 3.13 and 3.14 as well as in Equation 3.18 and 3.19.

3.2 Experimental Setup

3.2.1 Dataset

We use the LibriSpeech corpora (Panayotov et al. [2015]) for our experimentation. The LibriSpeech corpora are obtained from English audiobooks read by native English speakers. They are standard corpora used for the evaluation of ASR systems. The corpora consist of several subsets created based on the word error rate (WER). The WER is a quality measure for speech to text systems. It is computed by transcribing speech and comparing it with the reference transcripts obtained from the text of the book. In this work we use **train-clean-100** for the purpose of training and **test-clean-100** for the purpose of evaluation. The “clean” suffix is associated with the low WER implying the high quality of speech signals. In Table 3.1 we list the statistics of the dataset.

	Train	Test
Sampling Rate	16 kHz	16 kHz
Total Duration	100.6 hrs.	5.4 hrs
Per Spk. Duration	25 min.	8 min.
Female Speakers	125	20
Male Speakers	126	20
No. of Speakers	251	40

Table 3.1: Data Statistics of LibriSpeech Corpus

3.2.2 Preprocessing

The speech files in the LibriSpeech dataset are of varying length. To train our network we work with the fixed-length representation. We split each speech file into segments of 4 seconds. If any segment is smaller than 4 seconds we pad it with zeros in the end. This way, we end up with 48,301 samples of female and 48,139 samples of male voice data. We further apply the STFT transformation to obtain the time-frequency representation of all samples. For STFT we use the librosa Python package¹ (McFee et al. [2015]). The configuration used for the STFT is further listed in Table 3.2. Since speech signals are bandlimited signal, we trim the representation at the Nyquist frequency and pad the time axis. Finally, we obtain two matrix representations of shape (500, 256) of the magnitude spectrogram and phase spectrogram, where 500 is the number of time steps and 256 is the frequency bin.

In the first part of the experiment, we use the magnitude spectrogram on different scales. For the mel scale, we define 128 mel filters and apply them on the linear magnitude matrix. This provides the mel representation of shape (500, 128) where entries in columns are the response of respective mel filters. Since the transformation to the mel scale is not invertible we take pseudo-inverse of mel filterbank to approximate the inverse linear transformation, which interestingly has a negligible effect on the perceptual quality of speech. In the second part we use a full spectrogram representation. We transform the phase to the IF by first unwrapping the phase over the time axis and then taking the finite difference along the same axis. We represent the magnitude and IF spectrum together as an image with 2 channels of shape (500, 256, 2).

In the last layer of generator networks, we use `tanh` as a non-linearity. To match the output of `tanh` we scale the coefficient matrix to the range $[-1, 1]$. Unlike the image, the maximum value of spectrogram coefficients varies from waveform to waveform. To consider this effect we first scale each spectrogram with the maximum value and then scale it to range $[-1, 1]$. The scaling by the maximum also ensures the input samples have same loudness range.

Likewise, we also apply the above steps on our test set of 40 speakers present in

¹<https://github.com/librosa/librosa>

Parameters	Value
N-FFT	512
Hop-length	128
window	hanning
window-size	512

Table 3.2: STFT Configuration

`test-other-100`. We end up with 2,703 samples each of female and male speakers.

3.2.3 Network Details

The network architecture used in our experiments is based on UNIT (Zhu et al. [2017]). The two generator networks are implemented as VAEs given by pairs of an encoder and decoder network (E_x, G_x) and (E_y, G_y). The encoder network consists of three convolution layers, followed by four ResNet blocks. The first convolution layer consists of 32 filters of size 7×7 with a stride of 1. The next two layers consist of 64 and 128 filters of size 3×3 with a stride of 2. Each ResNet block consists of two convolution layers with 512 filters of size 3×3 and a stride of 1. Likewise in the decoder network, we have four ResNet blocks of the same configuration followed with three upsampling blocks with an upsampling factor of 2. After each upsampling block, there is a convolution layer. The three respectively, convolution layers have 128, 64 and 1 filters of size 5×5 , 5×5 and 7×7 and padding of size 2, 2 and 3. In all intermediate layers, after every convolution, we use a LeakyReLU as a non-linearity with a slope of 0.2. In the ResNet blocks, non-linearity is applied only after the first convolution layer. In the final layer, we use `tanh` as a non-linearity. Furthermore, to impose the shared space assumption discussed previously in Section 3.1, the last layer of E_x and E_y and the first layer of G_x and G_y are defined to share weights.

The discriminator network consists of 6 convolution layers. The first layer has 64 filters of size 3×3 and a stride of 2. In the next four layers, we double the number of filters after every layer and use a filter size of 3×3 and a stride of 2. All layers are followed by a LeakyReLU as a non-linearity with a slope of 0.2. The final layer performs a convolution with 1 filter of size 2×2 and a stride of 1.

The cross-entropy loss results in a vanishing gradient problem. Thus, to avoid it and to ensure a stable training, we use the LS-GAN objective (Mao et al. [2017]). Furthermore, the input to the discriminator is taken at two more scales obtained by downsampling the input by a factor of 2 and 4. The multi-scale input is helpful for the progressive learning of the discriminator (Wang et al. [2018]).

We refer to our base network as VC-GAN, the network with consistency loss discussed in Section 2.1.3.1 as VC-Con-GAN, the network with projection loss discussed in Section 3.1.3.1 as VC-GL-GAN. All our implementation was done using NAbra, a neural network framework developed by Sony². All our experiments were

²<https://nnabla.org/>

Spectrogram	Input Shape	Training Time hrs.	Inference Time sec.	No. of Parameters
Mag/LogMag	(500, 256, 1)	112.88	0.290	21,509,384
Mel/LogMel	(500, 128, 1)	49.17	0.239	21,509,384
MagIF/LogMagIF	(500, 256, 2)	232.45	0.179	21,509,384

Table 3.3: Network time and parameters comparison of VC-Con-GAN for different spectrogram representations.

Loss	Hyperparameter
Reconstruction Term	10
Translation Term	10
Cycle Term	10
KL for reconstruction	0.01
KL for cyclic	0.01
Magnitude consistency	0.0003
Full spectrogram consistency	0.005
w_1, w_2	0.65

Table 3.4: Hyperparameter Configuration where reconstruction implies ($X \rightarrow X$, $Y \rightarrow Y$), translation implies ($X \rightarrow Y$, $Y \rightarrow X$) and cycle implies ($X \rightarrow Y \rightarrow X$, $Y \rightarrow X \rightarrow X$).

executed on GeForce GTX 1080 Ti GPU with 11 GB of memory.

3.2.4 Training Procedure

We train all our network using the Adam optimizer (Kingma and Ba [2014]) with a learning rate of 0.0001 and the momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$. Due to the size of the network and the activations, we used a minibatch training with a batch size of 2 containing one image from domain X and one from domain Y . We decide the batch size based on the available GPU memory. For the mel scale, the input dimension is of smaller size and activations take less memory. Thus, we were able to fit a batch of size 6 with 3 samples of each domain.

In Table 3.3 we compare the training time, inference time and parameters of different input representations. Our network has a lot of hyperparameters. Due to the high computational cost, we avoid hyperparameter tuning. We use default configurations given in Table 3.3. For spectrogram consistency loss we decide the value of the hyperparameter by looking at the scale of other loss terms. In our experiments, we observed the spectrogram consistency loss becomes stagnant after a fixed number of iterations. This could be attributed to the convergence of the loss term. Therefore we decay the hyperparameter of the consistency loss by a factor of 0.9 after every 10,000 iterations.

To solve the optimization problem of our network we use an alternating gradient scheme as described in Goodfellow et al. [2014]. We first fix the discriminators D_1

and D_2 and apply a gradient descent to update the parameters of our two generator networks (E_1, G_1) and (E_2, G_2) . We then fix the generators and apply gradient ascent to update the parameters of D_1 and D_2 . We train all our networks for 1,000,000 iterations where each iteration refers to a minibatch update. After every 10,000 iterations, we decay weights by a factor of 0.0001 and the learning rate by a factor of 0.5.

3.2.5 Postprocessing

Previously, we applied a series of transformations to prepare the input representation of our network. To reconstruct the coherent waveform we further need to invert all transformations. In this section, we look at the series of steps required to reconstruct the time domain waveform of the speech signal.

In the first set of experiments, we rescale the generated spectrogram from the range $[-1, 1]$ to $[0, 1]$ for the magnitude and $[0, \log 2]$ for the log magnitude. After that in the case of the mel scale, we take the pseudo-inverse of the mel filterbank matrix and used it to transform the mel spectrogram to the linear magnitude scale. We then performed GLA on the linear magnitude spectrogram to reconstruct the time domain signal. In the second set of experiments, for the magnitude, similar to the previous case, we scale the spectrogram to $[0, 1]$, and to $[0, \log 2]$ for the log magnitude. For the IF we first take the finite summation along the time axis and then wrap it to lie in the range $(-\pi, \pi]$. Thus, we obtain the phase and linear magnitude spectrogram representations. We further combine them and apply the ISTFT transformation to reconstruct time domain waveform.

Chapter 4

Evaluation and Discussion

In the Chapter 3, we looked at the detailed formulation of our problem, network, and implementation. In this chapter, we will look at the details of the qualitative as well as quantitative evaluation of our framework. For evaluation we report the IS, FID and domain classification accuracy. For details on the IS and FID we refer to our previous discussion in Section 2.2.6.

Since the IS and FID require the feature space representation of the data, we developed a NN-based speaker identification network to map generated and test samples to the feature space. We then use the feature space to compute the FID which gives us a measure of the distance between the probability distribution of the test data and of the generated data. Furthermore, we use the same speaker identification network to compute the IS. In Section 4.1.1 we look at the details of the speaker identification network. We also evaluate the separability of the generated male and female voice samples. For this we first train a domain classifier and then use it to predict the domain of the generated samples. For a baseline, we compare the performance of the same classifier on held out test samples. In Section 4.1.2 we discuss the details of the domain classifier.

In Section 4.2 we discuss the results of our speech-to-speech synthesis network. We start our discussion with the first experimental setup in Section 4.2.1 where we only use the magnitude part of the time-frequency representation for training of our framework. We have different loss terms in our network, thus to understand the effect of each term on the quality of generated samples we further perform an ablation study in Section 4.2.1.1. In Section 4.2.2 we look at the results of the second setup with the full spectrogram generation. Finally, in Section 4.2 we conclude the chapter with the discussion on a perceptual evaluation performed by human subjects.

4.1 Evaluation Setup

4.1.1 Speaker Identification Network

The speaker identification is defined as the task of identifying the speaker based on the segment of a speech signal. Let K be the number of speakers and N be the

number of speech segments of K speakers. For each segment x_n , we apply the STFT transformation and take the magnitude spectrogram $\hat{X}_n$. For every speaker $k \in K$, the spectrogram $\hat{X} \in \mathbb{R}^{F \times T}$ is a time-frequency matrix where F is the number of frequency bins and T is the number of time steps. Let y be an indicator random variable modeling the probability of observing the k^{th} speaker defined as:

$$y_{nk} = \begin{cases} 1, & \text{if the speaker label for segment } n \text{ is } k \\ 0, & \text{otherwise} \end{cases}$$

Then the probability of a speaker k given a spectrogram of speech segment $\hat{X}_n$ is expressed as $p(y_{nk} = 1 | \hat{X}_n)$. We use the multi-class cross-entropy loss and define the objective of the speaker identification task as:

$$V = - \sum_{n=1}^N \sum_{k=1}^K y_{nk} \log(p(y_{nk} = 1 | \hat{X}_n)) \quad (4.1)$$

We train an end-to-end NN which takes the spectrogram representation of speech and predicts the log-likelihood probabilities of a speaker label. Our network is working at the frame level features and is based on the network proposed by [Okabe et al. \[2018\]](#). Specifically, the spectrogram matrix $\hat{X}_n = \{\hat{X}_n^{(1)}, \dots, \hat{X}_n^{(T)}\}$ is represented as a sequence of T frames, where each frame $\hat{X}_n^{(t)} \in \mathbb{R}^F$. The network consists of three parts:

- a time-delay neural network (TDNN) ([Waibel et al. \[1995\]](#)) with three one-dimensional convolutional layers to extract frame-level features, the first layer takes input frames at $\{t-2, t-1, t, t+1, t+2\}$, the second layer takes the output of the first layer at time steps $\{t-2, t, t+2\}$ and third layer takes the output of previous layer at $\{t-3, t, t+3\}$
- statistical pooling to combine frame-level features to obtain a fixed-length representation. The coefficients at frame-level features of certain frames are more relevant for discriminating speakers. To model the importance of these features we use an attention mechanism in the pooling operation. Specifically, for any segment $\hat{X}_n$, let h_n^t be the hidden frames after the last TDNN layer, let e_t be attention scores for each frame h_t defined as a parametric transformation

$$e_t = \text{softmax}(v^T \cdot \sigma(W h_t + b) + a) \quad (4.2)$$

where $v \in \mathbb{R}^F$ is a weight vector, $W \in \mathbb{R}^{F \times F}$ is a weight matrix and a, b are bias terms. The statistics $\hat{\mu}$ and $\hat{\sigma}$ of the attentive pooling is obtained as:

$$\hat{\mu} = \sum_{t=0}^T e_t \cdot h_t \quad (4.3)$$

$$\hat{\sigma} = \sqrt{\sum_{t=0}^T e_t h_t \odot h_t - \hat{\mu} \odot \hat{\mu}} \quad (4.4)$$

The fixed-length representation is obtained by concatenating $\hat{\mu}$ and $\hat{\sigma}$.

- two fully connected dense layers of dimensionality 512 and 256. The output of the final dense layer is interpreted as utterance level features modeling the speech variation of a speaker.

Finally, the utterance features are passed to a softmax layer to predict the log-likelihood probabilities of speakers. We use a LeakyReLU with a slope of 0.2 as a non-linearity after each layer. To prevent the network from overfitting we use dropout with a probability of 0.3 after each TDNN layer and 0.5 after each dense layer.

The above architecture learns utterance level features to discriminate between the speakers. Thus, by training a model on a large number of speakers we can use it to map an unseen speaker to the embedding space. For speech signals, the representations in the embedding space are interpreted as the phonetics and utterance properties of the speakers.

We use the **train-clean-100** data of the 251 speakers from the LibriSpeech corpus. For each speaker we randomly sample a set of 70% the recordings for training and the remaining 30% for the evaluation of the classifier. We use Adam with a learning rate of 0.001 and momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$. We use early stopping and use the classifier with the best accuracy on the 30% held out samples for feature extraction. The part (b) of Figure 4.2 and 4.1 shows the evolution of the loss values on the training and validation set. Table 4.1 describes the accuracy of the speaker identification network on the held out samples.

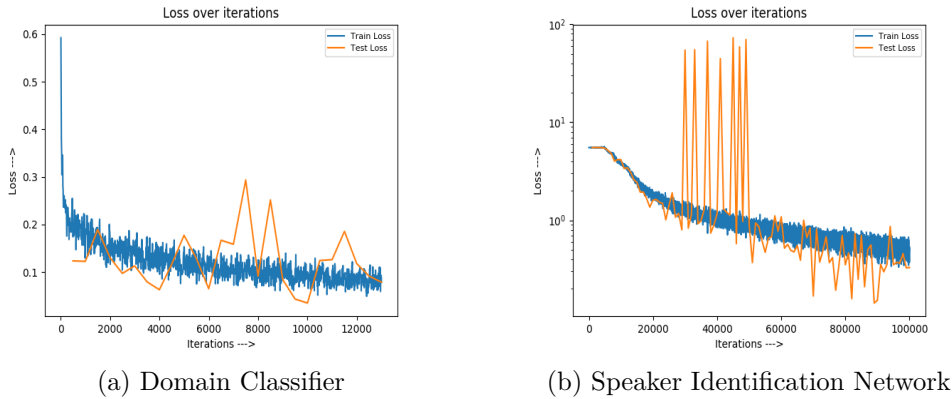


Figure 4.1: The loss plot of the domain classifier and speaker classifier trained on the LogMel spectrogram.

4.1.2 Domain Classification

To evaluate the quality of the generated samples we train a domain classifier network. The network is trained with an objective to discriminate between the voice of male and female speakers. We use the magnitude representation of the spectrogram for

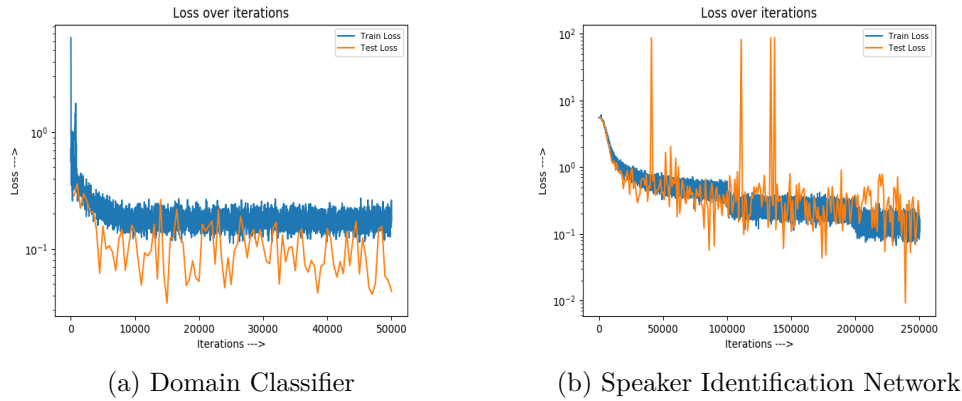


Figure 4.2: The loss plot of domain classifier and speaker classifier trained on Log-Mag spectrogram. For better resolution we scale the loss values to log scale.

Spectrogram	Test Accuracy	
	Domain classification	Speaker identification
LogMag	0.975	0.875
LogMel	0.968	0.967

Table 4.1: Evaluation of the domain classifier and speaker identification network. For the domain classifier test accuracy is reported on the held out set of 40 speakers. For the speaker identification network the test accuracy is reported on 30% held out speech samples of 251 speakers in `train-clean-100` set.

training. Let x be a speech segment, $\hat{X}$ be a magnitude spectrogram, let y_n be an indicator random variable defined as

$$y_n = \begin{cases} 1, & \text{if } n^{\text{th}} \text{ segment is a sample of female voice} \\ 0, & \text{if } n^{\text{th}} \text{ segment is a sample of male voice} \end{cases}$$

Then $p(y_n = 1|X_n)$ is the probability of a sample X_n belonging to the Female class. Conversely, $p(y_n = 0|X_n)$ is the probability of a sample X_n belonging to the male class. The learning objective for N segments of speech is defined as:

$$V = - \sum_{n=0}^N y_n \log(p(y_n = 1|X_n)) + (1 - y_n) \log(1 - p(y_n = 0|X_n)) \quad (4.5)$$

We implement the domain classifier as a 2 layered perceptron network with 512 hidden units in a first layer and 256 units in a second layer. After each layer, we use a LeakyReLU with a slope of 0.2 as a non-linearity. The final layer consists of a single neuron followed by a sigmoid function to output the likelihood probability of a class. We train the network using the Adam optimizer with a learning rate of 0.0001 and momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$. In Figure 4.2 and 4.1 part (a) the evolution of loss values on training and validation set is shown. Table 4.1 describes the accuracy of the domain classification network on the held out set.

4.1.3 Evaluation Measures

The evaluation of the generated speech samples is a difficult task. It is not clear how to define a performance measure which correlates to the perceptual quality of speech. Therefore, we perform a quantitative as well as qualitative evaluation of the generated samples. We report all measures on the held out set of 40 speakers.

4.1.3.1 Quantitative evaluation

We report the FID and IS to evaluate the diversity of the generated samples. We use a speaker identification network to map translated and real samples to the feature space. We extract the features after the penultimate layer of the speaker identification network. We then use these features to compute FID and IS. To get an estimate of the range of FID and IS we also report the scores on the untrained network which serves as a measure of worst-case scenario when the network learns nothing. Table 4.2 describes the values of the FID and IS for an untrained network. For details on FID and IS please refer to our previous discussion in Section 2.2.6.

We also report the domain classification accuracy which gives the measure of separability of the domain of male and female speakers. The high value of the domain accuracy means female-to-male translated samples are more like male and male-to-female translated samples are more like a female.

Spectrogram	FID		IS	
	<i>min.</i>	<i>untrained</i>	<i>untrained</i>	<i>max.</i>
LogMag	0	45140	1.02	5406
LogMel	0	25142	1.03	5406

Table 4.2: Empirical bound on FID and IS. *min.* stands for minimum value (is zero since FID is the distance), *max.* stands for maximum value (is equal to support test size for details refer to Heusel et al. [2017]) and *untrained* is obtained from randomly initialized network.

4.1.3.2 Qualitative evaluation

We use human subjects to get the evaluation of translated speech samples. We conduct two perceptual evaluation test. In the first test, we ask listeners to rate the quality of translated speech samples on a scale: Excellent (80-100), Good (60-80), Fair (40-60), Poor (20-40) and Bad (0-20). In the second test, we ask listeners to rate the extent to which each translated sample sound like a female or male person. For this we use a scale: Female (80-100), Somewhat Female (60-80), Not sure (40-60), Somewhat Male (60-80) and Male (80-100). We use webMUSHRA (Schoeffler et al. [2018]) to set up the online human evaluation test. We report the mean opinion score (MOS) for each model. The MOS is defined as an arithmetic mean of over single rating perform by human evaluators:

$$\text{MOS} = \frac{\sum_{i=1}^N r_i}{N} \quad (4.6)$$

where r_i is a rating by an individual participant and N is the number of participants. We use a held out set of 40 speakers for evaluation. We translate all the samples in the held out set and randomly select a subset of 100 samples. From this subset, we arbitrarily select 11 samples of male speakers and 11 samples of female speakers. For comparison in our listening test, we present listeners with the translation of the same sample from different models. Figure 4.3 describe the interface of the listening test.

4.1.4 Robustness of FID and IS

To further interpret the FID and IS scores we perform the robustness analysis. We add increasing disturbance to generated samples and see the effect on FID and IS. We apply the following disturbance on the spectrogram:

- **Gaussian noise:** For a spectrogram $\hat{X}$, we compute a noisy version $(1 - \alpha)\hat{X} + \alpha\eta$, where $\eta \sim \mathcal{N}(0, 1)$ is sampled from Gaussian noise with zero mean and unit variance where α is a parameter to control the mix of signal and noise. This noise has a degrading effect on the quality of the speech signal. A higher fraction of noise results in low signal to noise ratio.



Figure 4.3: An example of the GUI we used for the perceptual evaluation. In the domain evaluation for male speakers we subtract reported scores from 100. This way we end up with the same scale for male and female speakers.

- **Salt and pepper noise:** For a spectrogram $\hat{X}$ we randomly flip the coefficient values with a probability α . We set the coefficient value to 0 or 1 with a probability of 0.5. This results in peaks in the spectrogram representation. These peaks affect the perceptual quality of sound giving the robotic nature to the sound.

α (Noise parameter)	FID	IS
0	28.57	19.86
0.1	192.49	23.73
0.25	602.18	81.16
0.50	1715.17	1898.26
0.75	3567.12	1823.59

Table 4.3: FID and IS score with increasing Gaussian noise for the VC-Con-GAN. Here we consider a joint distribution of male and female speakers.

Table 4.3 and 4.4 describe the FID and IS score for different values of the noise factor α . We can see the FID score is consistent with the increasing noise. On the other hand, the IS has a strange behavior since it increases for Gaussian noise and fluctuates for salt and pepper noise. The robustness of the FID score for Gaussian noise implies the metric is correlated to signal to noise ratio and the salt and pepper noise indicates that the FID is also robust to capture a difference between natural and robotic sounding voice. For the above reason, we will build our discussion on

α (Noise parameter)	FID	IS
0	28.57	19.86
0.1	1466.67	631.03
0.25	5029.97	1.18
0.50	11691.86	1.04
0.75	16804.00	1.03

Table 4.4: FID and IS score with increasing salt and pepper noise for the VC-Con-GAN. Here we consider a joint distribution of male and female speakers.

the FID. For the sake of completeness, we also report the IS but we do not use it for further discussions.

4.2 Results and Discussion

In this section, we look at the results and compare the performance of the different models. We refer to the base UNIT network as VC-GAN, the network with the consistency loss discussed in Section 2.1.3.1 as VC-Con-GAN, and the network with the projection-based consistency loss discussed in Section 3.1.3.1 as VC-GL-GAN.

4.2.1 Magnitude spectrogram generation

In this section, we focus our discussion on the first setup where we only use the magnitude part of the STFT representation. Table 4.5 compares the performance of the VC-GAN and VC-Con-GAN for different magnitude scales. For the VC-Con-GAN we train the network only on the magnitude and log-magnitude scale and omit on the mel scale. This is because the consistency constraint of the spectrogram only holds on these two representations.

We separately report the FID scores for female and male domain. We observed the log transformation generally helps the training and provides lower FID scores. In the spectrogram representation, the low-frequency components are associated with the information and the high-frequency components are associated with noise. The log transformation spread the spectrogram by enhancing the lower components and compressing the noise part. We attribute the better performance of the log scale to the enhancement of speech information which makes it easier for NNs to learn meaningful features.

The VC-Con-GAN outperforms the VC-GAN in terms of FID scores. The best value of the FID is achieved for the log magnitude representation. The introduction of the consistency loss provides an improvement over the base model.

In Figure 4.4 we show an example of a spectrogram of a four second segment of the original speaker and the corresponding translated version. We show the plot of our best model VC-Con-GAN trained on the log magnitude spectrogram. We can

see the network learn to shift the tones as well as learn to change the shape and spacing of contours to translate the voice.

Method	Spectrogram	IS	FID		Accuracy
			Female	Male	
VC-GAN	Mag	22.14	130.32	100.36	0.894
	Mel	30.64	123.79	89.16	0.935
	LogMag	21.35	76.90	74.18	0.935
	LogMel	32.06	127.14	116.13	0.938
VC-Con-GAN	Mag	22.89	94.83	83.52	0.928
	LogMag	19.86	70.11	69.50	0.860

Table 4.5: Evaluation of generated samples of different GANs trained on Magnitude spectrogram. We compare IS, FID and domain classification accuracy. We use held out set of 40 speakers for evaluation.

4.2.1.1 Ablation Study

We have several loss functions in our optimization problem. To understand the direct effect of each term on the quality of the generated samples, we perform an ablation study. In this study, we remove the respective loss term and observe the effect on the FID and domain classification accuracy. Table 4.6 describes the effect on FID scores and Table 4.7 describes the effect of loss terms on domain accuracy. The removal of the spectrogram consistency loss results in an increase of FID scores for the magnitude as well as the log magnitude scale. We observed that the effect on the domain accuracy is not consistent. It decreases for the magnitude and increases for the log magnitude. The explanation for this is not clear. We think this also depends on the generalization of the base domain classifier. Henceforth, for the evaluation of the generated samples, we do not rely on the classification accuracy. We use it as a sanity measure to check whether voice samples are translated across the domain. We will later see in Section 4.2.3 the FID scores are also consistent with the human evaluation. Thus, we conclude that the FID is a better measure for the evaluation.

The other loss terms follow from the base UNIT network and have a varying effect for different input representations. The KL loss term in the VAE does not provide any improvement in terms of the FID. The cycle consistency term is useful for the magnitude representation but not for the mel, log magnitude, and log mel representation. We claim this is due to the weight sharing condition which also ensures a pair of corresponding samples in the two domains are represented by the same latent features. To support our claim we removed the cycle consistency loss, as well as the weight sharing. In this case, we observed that all networks fail at the task of domain translation.

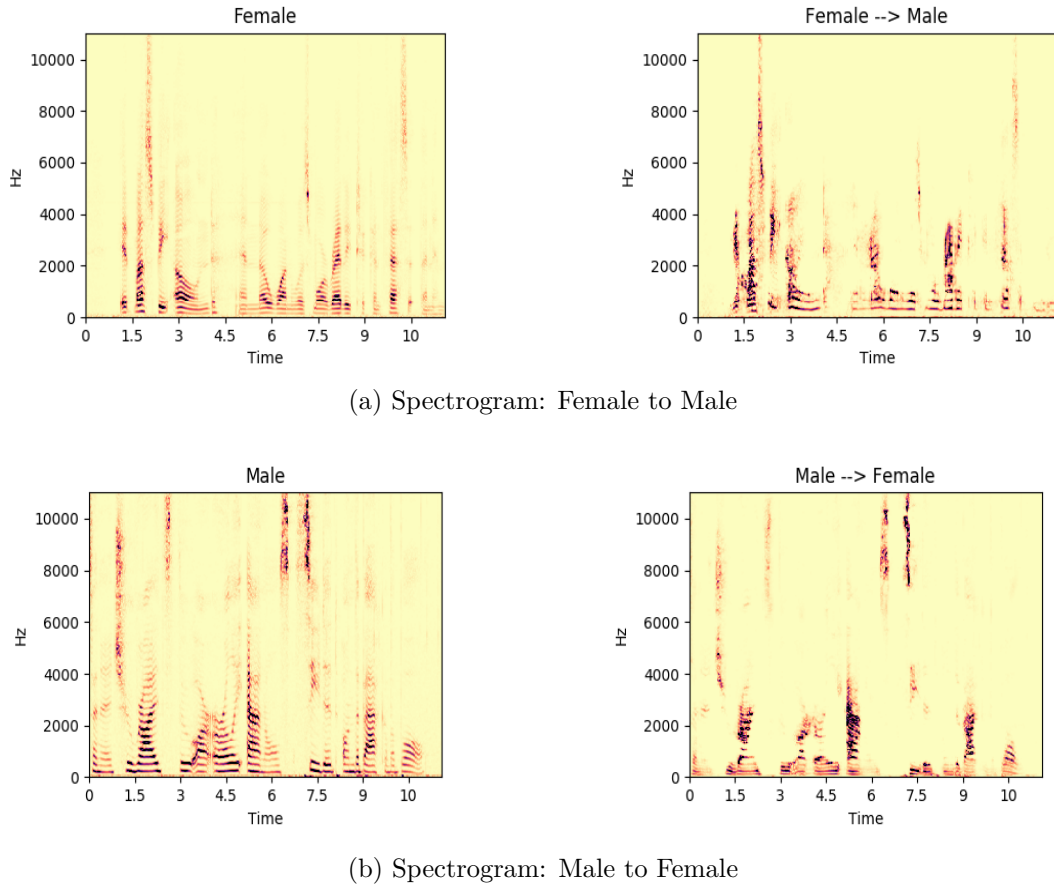


Figure 4.4: An example of spectrogram translation from female to male at the top and male to female at the bottom.

Spectrogram	Full Loss	No Con	No VAE	No CC	No Shared-CC
Mag	0.928	0.894	0.889	0.500	0.499
Mel	-	0.935	0.906	0.953	0.000
LogMag	0.860	0.935	0.928	0.726	0.499
LogMel	-	0.938	0.939	0.909	0.500

Table 4.6: Effect of different loss terms on the domain accuracy where full loss represent our VC-Con-GAN model, NoCon is obtained by removing the spectrogram consistency loss, No VAE is obtained by removing the VAE loss, No CC is obtained by removing the cycle consistency loss term and No shared-CC is obtained by removing the weight sharing condition and the cycle consistency loss.

Spectrogram	Full Loss		No Con		No VAE		No CC		No Shared-CC	
	Female	Male	Female	Male	Female	Male	Female	Male	Female	Male
Mag	94.83	83.52	130.32	100.36	87.08	81.57	2491.03	1273.12	408.45	4209.07
Mel	-	-	123.79	89.28	139.22	91.99	103.83	112.37	1546.00	779.61
LogMag	70.11	69.50	76.90	74.18	74.88	69.14	94.71	108.09	420.11	730.16
LogMel	-	-	127.14	116.01	94.31	98.51	122.67	97.74	2310.52	461.62

Table 4.7: Effect of different loss terms on FID scores where full loss represents our VC-Con-GAN model, NoCon is obtained by removing the spectrogram consistency loss, No VAE is obtained by removing the VAE loss, No CC is obtained by removing the cycle consistency loss term and No CC-VAE is obtained by removing the weight sharing, VAE as well as the cycle consistency loss term.

4.2.2 Full spectrogram generation

The quantitative evaluation of this setup is difficult. It is mainly due to the difficulties associated with obtaining the feature space representation of the full spectrogram. The speaker identification network discussed in Section 4.1.1 fails to learn from the full spectrogram. We think an additional phase part serves as noise and introduces additional difficulties in the learning task since the speaker identification task relies on features which are dependent on phonetic variations or the accent of a speaker and these features can be independently obtained from the magnitude spectrogram. Thus for quantitative evaluation, we discard the phase part and only use the magnitude part to get the feature representation to compute the FID and IS. Likewise, we only use the magnitude part to report the domain classification accuracy. Since we only use the magnitude spectrogram for the evaluation we don't rely on these results and only report them for the sake of completeness. For this part we mainly focus on the perceptual evaluation discussed in following section. Table 4.8 describes the performance of this setup.

Method	Spectrogram	IS	FID		Accuracy
			Female	Male	
VC-GAN	MagIF	19.73	81.13	65.34	0.819
	LogMagIF	15.33	70.58	57.01	0.783
VC-Con-GAN	MagIF	19.46	74.14	78.03	0.868
	LogMagIF	17.66	59.38	75.40	0.843
VC-GL-GAN	MagIF	19.56	71.57	77.76	0.933
	LogMagIF	18.60	62.25	73.26	0.880

Table 4.8: Evaluation of generated samples of different GANs trained on the full-spectrogram. We compare the Inception Score (IS), Fréchet Distance (FID) and domain detection accuracy. We report the accuracy on unseen original test samples and generated samples.

4.2.3 Human Evaluation

For the qualitative evaluation of each model, we select the spectrogram with the best FID scores. We do the perceptual evaluation of these models using the scales discussed in Section 4.1.3.2. There were 28 participants in the study. Table 4.9 compares the MOS of the different models. Figure 4.5 compares the histogram of the different models with 95% confidence interval.

The consistency loss with the log magnitude representation performs best in terms of the quality as well as the domain translation. Furthermore, the results are also consistent with the FID scores. For the second setup, we observed the results are not good in terms of the quality and much worse than the magnitude-based representations. But in terms of the domain translation the results are comparable.

The two different consistency terms used in this case provide an improvement over the base model. The consistency of the magnitude spectrogram performs better than the projection-based consistency condition of the full spectrogram. The explanation is not clear and we will look into it in the future work.

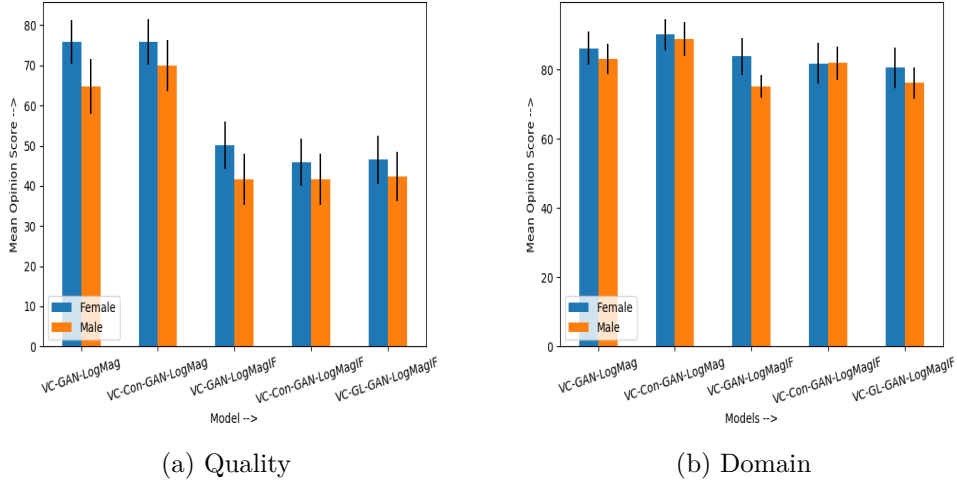


Figure 4.5: Mean opinion score for the perceptual listening test

Method	Spectrogram	MOS			
		Quality		Domain	
		Male	Female	Male	Female
VC-GAN	LogMag	64.66	76.28	82.62	87.18
	LogMagIF	45.46	41.27	74.43	83.22
VC-Con-GAN	LogMag	70.13	76.29	88.44	89.50
	LogMagIF	41.32	49.81	80.98	81.17
VC-GL-GAN	LogMagIF	42.19	46.43	75.35	79.81

Table 4.9: Mean opinion scores

We further performed a Wilcoxon signed-rank test (Wilcoxon et al. [1970]) to see the significance of the difference between VC-GAN and VC-Con-GAN. In particular, we evaluate whether the MOS of the two models for the same sample is similar. For the domain evaluation test, we get a p-value of 0.001865 for female-to-male samples and p-value of 0.000181 for male-to-female samples. This indicates the distribution of scores of the two models is significantly different. Likewise, we also performed the test for comparing the quality of samples of the two models. We get a p-value of 0.000078 for female-to-male samples and a p-value of 0.476942 for male-to-female speakers. This implies that the quality of the two models is comparable for male-to-female speakers but significantly different for female-to-male speakers. Together with the Wilcoxon test and the MOS, we conclude the VC-Con-GAN outperforms

the VC-GAN for the domain translation task. Furthermore, it is also a clear winner for the quality of the generated male voice and is comparable to the VC-GAN for the quality of the generated female voice.

Chapter 5

Conclusion and Future Research

In this thesis, we introduced a generative adversarial network based model for unsupervised cross-domain speech-to-speech synthesis. We demonstrated the performance of our model on the task of translating the voice of male speakers to the voice of female speakers and vice versa. Our model utilizes the time-frequency representation of speech signals. The time-frequency representation of speech signals is obtained using a short-time Fourier transform (STFT) operation. In contrast to time-amplitude representation, the time-frequency representation is robust for the generation of high-quality speech signals. However, the choice of the time-frequency representation poses the challenge of the spectrogram consistency, that is finding a coherent time-domain waveform, which is the target for the task of speech synthesis. We outlined a comprehensive discussion on the consistency of the time-frequency representation. We presented two experimental setups to address the problem of coherent waveform reconstruction. In the first setup, we ignore the phase spectrogram and only utilize the magnitude spectrogram as an input to our model. Furthermore, we utilize the consistency condition of the magnitude-spectrogram to define a loss term to constrain the model such that translated spectrograms (male-to-female or female-to-male) have a valid time-domain waveform. During inference time we use the Griffin-Lim algorithm to reconstruct a coherent time-domain waveform from the translated magnitude-spectrogram, thus, accomplishing our goal of speech-to-speech synthesis. In the second setup, we utilize the full time-frequency representation as an input to our model. We transform the phase spectrogram to the instantaneous frequency (IF) representation. The IF is free of discontinuities present in the phase spectrogram which we hypothesize should make it easier for the model to learn meaningful features. In this setup, we utilize the consistency condition of the full time-frequency representation by performing an iteration of the STFT and inverse-STFT (ISTFT) operation on the translated spectrogram. Finally, at inference time to perform speech synthesis we apply the ISTFT operation on the translated spectrograms.

We performed the quantitative as well as a qualitative evaluation of the different models. We report three quantitative scores, namely the inception score (IS), the Fréchet inception distance (FID) and the domain classification accuracy which

capture different facets of the model performance. We further show that the FID is a more reliable score since it is robust to different kinds of noise disturbances relating to the perceptual nature of sound. We performed extensive experiments based on different scales of the spectrogram representation. We showed the log scaling performs best in terms of the quality of the speech waveform. In the first setup, our empirical results demonstrate the importance of the spectrogram consistency constraint. For our consistency based model (VC-Con-GAN), we achieved lower a FID and the best MOS compared across all models. Furthermore, we performed a significance test and show the consistency based model is a clear winner. The results of our second setup are better in terms of the FID score but are not good in terms of the perceptual quality of speech. We think this could be improved by finding the right combination of hyper-parameters of the consistency loss term. Due to the limited time and high computational costs, we leave it as the future work.

Our work offers several promising research directions. In future work, we would like to investigate the methods for the latent space disentanglement of an input speech spectrogram. Specifically, we would like to learn two unique representations of an input speech signal. One representation to encode domain-specific features like phonetics, tone variations of speakers, and the other representation to encode domain-independent information like the linguistic features. By disentangling, the joint space into a domain-specific and a domain-independent space we can precondition the generation process on the specific speaker identifier. This will allow us to translate the voice of an input speaker to the desired target speaker. Such an application can be used to investigate the robustness of automatic-speech-recognition (ASR) systems. It can be used to demonstrate the threat to an ASR system in form of adversarial fake samples generated by the model. Besides this, our work can be used to introduce voice privacy in the online interview screening process. This will prevent discrimination based on the gender or race of an interviewee. The interviewee could select a required domain and a speaker identifier to convert their voice, and the interviewer will solely make the decision based on the performance.

Finally, we hope our work will increase the possibilities of time-frequency based GAN models for the adversarial generation of speech signals. Furthermore, we are optimistic that the work will open up the application to domain translation tasks for other audio signals, for instance in music genres translating from guitar to violin and many more.

Bibliography

- Arjovsky, M., S. Chintala, and L. Bottou
2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875*.
- Auger, F., É. Chassande-Mottin, and P. Flandrin
2012. On phase-magnitude relationships in the short-time fourier transform. *IEEE Signal Processing Letters*, 19(5):267–270.
- Bengio, Y., E. Laufer, G. Alain, and J. Yosinski
2014. Deep generative stochastic networks trainable by backprop. In *International Conference on Machine Learning*, Pp. 226–234.
- Denton, E. L., S. Chintala, R. Fergus, et al.
2015. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, Pp. 1486–1494.
- Doersch, C.
2016. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.
- Donahue, J., Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell
2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, Pp. 647–655.
- Duchi, J., E. Hazan, and Y. Singer
2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.
- Engel, J., K. K. Agrawal, S. Chen, I. Gulrajani, C. Donahue, and A. Roberts
2019. Gansynth: Adversarial neural audio synthesis. *arXiv preprint arXiv:1902.08710*.
- Gabor, D.
1946. Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, 93(26):429–441.
- Goodfellow, I.
2016. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

- Goodfellow, I., Y. Bengio, and A. Courville
2016. *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio
2014. Generative adversarial nets. In *Advances in neural information processing systems*, Pp. 2672–2680.
- Griffin, D. and J. Lim
1984. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243.
- Harris, F. J.
1978. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, 66(1):51–83.
- He, K., X. Zhang, S. Ren, and J. Sun
2016a. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Pp. 770–778.
- He, K., X. Zhang, S. Ren, and J. Sun
2016b. Identity mappings in deep residual networks. In *European conference on computer vision*, Pp. 630–645. Springer.
- Heideman, M., D. Johnson, and C. Burrus
1984. Gauss and the history of the fast fourier transform. *IEEE ASSP Magazine*, 1(4):14–21.
- Heusel, M., H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter
2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, Pp. 6626–6637.
- Hinton, G. E.
2009. Deep belief networks. *Scholarpedia*, 4(5):5947.
- Hornik, K., M. Stinchcombe, and H. White
1989. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.
- Hosseini-Asl, E., Y. Zhou, C. Xiong, and R. Socher
2018. A multi-discriminator cyclegan for unsupervised non-parallel speech domain adaptation. *arXiv preprint arXiv:1804.00522*.
- Huang, W.-C., H.-T. Hwang, Y.-H. Peng, Y. Tsao, and H.-M. Wang
2018. Voice conversion based on cross-domain features using variational auto encoders. In *2018 11th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, Pp. 51–55. IEEE.

- Hubbard, B. B.
1998. *The world according to wavelets: the story of a mathematical technique in the making*. AK Peters/CRC Press.
- Jaganathan, K., Y. C. Eldar, and B. Hassibi
2015. Phase retrieval: An overview of recent developments. *arXiv preprint arXiv:1510.07713*.
- Kahng, M., N. Thorat, D. H. P. Chau, F. B. Viégas, and M. Wattenberg
2018. Gan lab: Understanding complex deep generative models using interactive visual experimentation. *IEEE transactions on visualization and computer graphics*, 25(1):1–11.
- Kaneko, T. and H. Kameoka
2017. Parallel-data-free voice conversion using cycle-consistent adversarial networks. *arXiv preprint arXiv:1711.11293*.
- Karras, T., T. Aila, S. Laine, and J. Lehtinen
2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*.
- Kingma, D. P. and J. Ba
2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Kingma, D. P. and M. Welling
2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.
- Kopparapu, S. K. and M. Laxminarayana
2010. Choice of mel filter bank in computing mfcc of a resampled speech. In *10th International Conference on Information Science, Signal Processing and their Applications (ISSPA 2010)*, Pp. 121–124. IEEE.
- Krizhevsky, A., I. Sutskever, and G. E. Hinton
2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, Pp. 1097–1105.
- Kullback, S. and R. A. Leibler
1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.
- Le Roux, J., H. Kameoka, N. Ono, and S. Sagayama
2010. Fast signal reconstruction from magnitude stft spectrogram based on spectrogram consistency. In *Proc. Int. Conf. Digital Audio Effects*, volume 10.
- LeCun, Y., Y. Bengio, et al.
1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

- Liu, M.-Y., T. Breuel, and J. Kautz
2017. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, Pp. 700–708.
- Mao, X., Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley
2017. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, Pp. 2794–2802.
- Marafioti, A., N. Holighaus, N. Perraudin, and P. Majdak
2019. Adversarial generation of time-frequency features.
- McFee, B., C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto
2015. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, volume 8.
- Mitchell, T., B. Buchanan, G. DeJong, T. Dietterich, P. Rosenbloom, and A. Waibel
1990. Machine learning. *Annual review of computer science*, 4(1):417–433.
- Nakashika, T., T. Takiguchi, Y. Minami, T. Nakashika, T. Takiguchi, and Y. Minami
2016. Non-parallel training in voice conversion using an adaptive restricted boltzmann machine. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 24(11):2032–2045.
- Nyquist, H.
1924. Certain factors affecting telegraph speed. *Transactions of the American Institute of Electrical Engineers*, 43:412–422.
- Okabe, K., T. Koshinaka, and K. Shinoda
2018. Attentive statistics pooling for deep speaker embedding. *arXiv preprint arXiv:1803.10963*.
- Oord, A. v. d., S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu
2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.
- Panayotov, V., G. Chen, D. Povey, and S. Khudanpur
2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Pp. 5206–5210. IEEE.
- Perraudin, N., P. Balazs, and P. L. Søndergaard
2013. A fast griffin-lim algorithm. In *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Pp. 1–4. IEEE.

- Petzka, H., A. Fischer, and D. Lukovnicov
2017. On the regularization of wasserstein gans. *arXiv preprint arXiv:1709.08894*.
- Prša, Z., P. Balazs, and P. L. Søndergaard
2017. A noniterative method for reconstruction of phase from stft magnitude. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(5):1154–1164.
- Qian, K., Y. Zhang, S. Chang, X. Yang, and M. Hasegawa-Johnson
2019. Autovc: Zero-shot voice style transfer with only autoencoder loss. In *International Conference on Machine Learning*, Pp. 5210–5219.
- Qian, N.
1999. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151.
- Ratliff, L. J., S. A. Burden, and S. S. Sastry
2013. Characterization and computation of local nash equilibria in continuous games. In *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Pp. 917–924. IEEE.
- Salakhutdinov, R.
2015. Learning deep generative models. *Annual Review of Statistics and Its Application*, 2:361–385.
- Salakhutdinov, R. and G. Hinton
2009. Deep boltzmann machines. In *Artificial intelligence and statistics*, Pp. 448–455.
- Salakhutdinov, R., A. Mnih, and G. Hinton
2007. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, Pp. 791–798. ACM.
- Salimans, T., I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen
2016. Improved techniques for training gans. In *Advances in neural information processing systems*, Pp. 2234–2242.
- Schoeffler, M., S. Bartoschek, F.-R. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre
2018. webmushra—a comprehensive framework for web-based listening tests. *Journal of Open Research Software*, 6(1).
- Stevens, S. S. and J. Volkman
1940. The relation of pitch to frequency: A revised scale. *The American Journal of Psychology*, 53(3):329–353.

- Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang
1995. Phoneme recognition using time-delay neural networks. *Backpropagation: Theory, Architectures and Applications*, Pp. 35–61.
- Wang, T.-C., M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro
2018. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Pp. 8798–8807.
- Wilcoxon, F., S. Katti, and R. A. Wilcox
1970. Critical values and probability levels for the wilcoxon rank sum test and the wilcoxon signed rank test. *Selected tables in mathematical statistics*, 1:171–259.
- Zhu, J.-Y., T. Park, P. Isola, and A. A. Efros
2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, Pp. 2223–2232.