UNIVERSITY OF CALIFORNIA SAN DIEGO

Synthesis and Robust Detection of AI-generated Media

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Paarth Neekhara

Committee in charge:

Professor Julian McAuley, Chair
Professor Taylor Berg-Kirkpatrick
Professor Gary Cottrell
Professor Shlomo Dubnov

2023

The Dissertation of Paarth Neekhara is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

To my family.

EPIGRAPH

We are entering a new era of media manipulation, where AI-generated content blurs the lines between reality and fiction.

— ChatGPT

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would first like to express my heartfelt gratitude to my advisor and committee chair Julian McAuley, for his unwavering support and guidance throughout my PhD and for taking me under his wing when I casually walked into his office asking him to be my PhD advisor. Words cannot adequately convey the profound impact he has had on my life, and this dissertation would not have been possible without his steadfast belief in me. Secondly, I would like to thank my co-advisor Shlomo Dubnov who inspired me to conduct my current research in speech processing systems and has always helped shape rough ideas into concrete and thorough research.

I am indebted to Professor Gary Cottrell, who appointed me as a Teaching Assistant in his Neural Networks course when I was a Master's student still trying to figure out my research area. His advice has been invaluable for many of my research projects including some of my first machine learning papers, and his encouragement to this day motivates my career path. I would also like to thank Professor Taylor for serving on my PhD committee and for his constructive feedback on my research. I would also like to extend my sincere appreciation to my mentors at NVIDIA and Facebook AI: Boris Ginsburg, Cristian Canton, Jason Li and Jocelyn Huang for their valuable guidance during my internships.

I cannot overstate my gratitude to Shehzeen Hussain with whom I have collaborated on many research projects during my PhD. None of the work included in this dissertation would have been possible without her ideas, support and persistence throughout the research process.

My deepest thanks to my parents, my sister and the Blue Water family. Special thanks to Ankit Saxena who has always been a pleasure to work with and also a great friend. Thanks to all my friends in San Diego and around the globe who have been by my side throughout the process (to name a few): Palash Agrawal, Prakhar Pandey, Jenish Rakholiya, Vikas Yadav, Subodh Sonar, Shubham Singh, Shubham Pandey and Taruj.

Thank you to the following collaborators whose contribution was also crucial to my research included in this dissertation: Farinaz Koushanfar, Ryan Kastner, Mojan Javaheripi, Xinqiao Zhang, Nojan Sheybani, Javier Duarte, Malhar Jere, Jinglong Du, Joanna Bitton, Brian

Dolhansky, Miller Puckette.

Chapter 2 contains material found in the following two papers. (1) *Expressive Neural Voice Cloning*. 2021. Neekhara, Paarth; Hussain, Shehzeen; Dubnov, Shlomo; Koushanfar, Farinaz; McAuley, Julian. Asian Conference on Machine Learning 2021. (2) *Controllable Speech Synthesis with Iterative Refinement using Self Transformations*. 2023. Neekhara, Paarth; Hussain, Shehzeen; Ranjan, Rishabh; Dubnov, Shlomo; Koushanfar, Farinaz; McAuley, Julian. Currently under review for publication. The dissertation author was the primary investigator and author of these papers.

Chapter 3 contains material found in the following two papers. (1) *Adversarial Deepfakes: Evaluating Vulnerability of Deepfake Detectors to Adversarial Examples*. 2021. Hussain, Shehzeen; Neekhara, Paarth; Jere, Malhar; Koushanfar, Farinaz; McAuley, Julian. Winter Conference on Applications of Computer Vision 2021. (2) *Exposing Vulnerabilities of Deepfake Detection Systems with Robust Attacks*. 2023. Hussain, Shehzeen; Neekhara, Paarth; Dolhansky, Brian; Bitton, Joanna; Canton, Cristian; McAuley, Julian; Koushanfar, Farinaz. ACM Journal on Digital Threats: Research and Practice, Vol 3, 2022. The dissertation author and Shehzeen Hussain made equal contributions to this work.

Chapter 4 is a reprint of the material as it appears in *FaceSigns: Semi-Fragile Neural Watermarks for Media Authentication and Countering Deepfakes*. 2023. Neekhara, Paarth; Hussain, Shehzeen; Zhang, Xinqiao; Koushanfar, Farinaz; McAuley, Julian. Currently under review for publication. The dissertation author was the primary investigator and author of this paper.

VITA

2017 B.Tech, Indian Institute of Technology, Roorkee

2019 MS in Computer Science, University of California San Diego

2023 Ph.D in Computer Science, University of California San Diego

PUBLICATIONS

PUBLICATIONS

Shehzeen Hussain*, **Paarth Neekhara***, Jocelyn Huang, Jason Li, Boris Ginsburg "ACE-VC: Adaptive and Controllable Voice Conversion using Explicitly Disentangled Self-supervised Speech Representations", to appear in proceedings of *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing* (ICASSP), 2023

Shehzeen Hussain, Todd Huster, Chris Mesterharm, **Paarth Neekhara**, Farinaz Koushanfar "ReFace: Adversarial Transformation Networks for Real-time Attacks on Face Recognition Systems", to appear in proceedings of *2023 IEEE International Conference on Dependable Systems and Networks* (DSN), 2023

Shehzeen Hussain*, Nojan Sheybani*, **Paarth Neekhara***, Xinqiao Zhang, Javier Duarte, Farinaz Koushanfar "FastStamp: Accelerating Neural Steganography and Digital Watermarking of Images on FPGAs", in Proceedings of *2022 IEEE International Conference on Computer-Aided Design* (ICCAD), 2022

Shehzeen Hussain*, **Paarth Neekhara***, Brian Dolhansky, Joanna Bitton, Cristian Canton Ferrer, Julian McAuley, Farinaz Koushanfar "Exposing Vulnerabilities of Deepfake Detection Systems with Robust Attacks", in Proceedings of *2022 ACM Journal on Digital Threats Research and Practices* (DTRAP), 2022

Paarth Neekhara*, Shehzeen Hussain*, Jinglong Du, Shlomo Dubnov, Farinaz Koushanfar, Julian McAuley "Cross-modal Adversarial Reprogramming", in Proceedings of *2022 IEEE Winter Conference on Applications of Computer Vision* (WACV), 2022

Paarth Neekhara*, Shehzeen Hussain*, Shlomo Dubnov, Farinaz Koushanfar, Julian McAuley "Expressive Neural Voice Cloning", in Proceedings of *2021 Asian Conference on Machine Learning* (ACML), 2021

Paarth Neekhara, Brian Dolhansky, Joanna Bitton, Cristian Canton Ferrer "Adversarial Threats to DeepFake Detection: A Practical Perspective", in Proceedings of *IEEE/CVF Conference on*

Computer Vision and Pattern Recognition (CVPR) Workshops, 2021

Shehzeen Hussain*, **Paarth Neekhara***, Shlomo Dubnov, Julian McAuley, Farinaz Koushanfar "WaveGuard: Understanding and mitigating audio adversarial examples", in Proceedings of *2021 USENIX Security* (USENIX), 2021

Shehzeen Hussain*, **Paarth Neekhara***, Malhar Jere, Julian McAuley, Farinaz Koushanfar "Adversarial DeepFakes: Evaluating Vulnerability of Deepfake Detectors to Adversarial Examples", in Proceedings of *2021 IEEE Winter Conference on Applications of Computer Vision* (WACV), 2021

Paarth Neekhara, Shehzeen Hussain, Shlomo Dubnov, Farinaz Koushanfar "Adversarial Reprogramming of Text Classification Neural Networks", in Proceedings of *2019 Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2019

Paarth Neekhara*, Shehzeen Hussain*, Prakhar Pandey, Shlomo Dubnov, Julian McAuley, Farinaz Koushanfar "Universal Adversarial Perturbations for Speech Recognition Systems", in Proceedings of *Interspeech 2019*

Paarth Neekhara*, Chris Donahue*, Miller Puckette, Shlomo Dubnov, Julian McAuley "Expediting TTS Synthesis with Adversarial Vocoding", in Proceedings of *Interspeech 2019*

Shehzeen Hussain, Mojan Javaheripi, **Paarth Neekhara**, Ryan Kastner, Farinaz Koushanfar "FastWave: Accelerating Autoregressive Convolutional Neural Networks on FPGA", in Proceedings of *2019 IEEE International Conference on Computer-Aided Design* (ICCAD), 2019

ABSTRACT OF THE DISSERTATION

Synthesis and Robust Detection of AI-generated Media

by

Paarth Neekhara

Doctor of Philosophy in Computer Science

University of California San Diego, 2023

Professor Julian McAuley, Chair

Deep neural networks (DNNs) have enabled the creation of high-quality synthetic media. The intent of generating synthetic videos can be harmless as they can be used for tasks such as advertisement campaigns, and face replacement in movies and animated films. However, DNNs can also be trained to synthesize facially manipulated videos called Deepfakes, which may be used maliciously to defame famous personalities, spread misinformation and influence elections based on false facts.

My research focuses on the responsible use of deep learning for media synthesis. On the synthesis side, I develop methods for natural and expressive speech synthesis methods for new speakers in data-limited settings. My work enables the creation of a digital voice clone of

a person that can either be generated using text or a reference speech from a different speaker. Another area of my research focuses on the robust detection of Deepfake videos. We investigate the limitations of current DeepFake detection methods and demonstrate that they can be easily bypassed using adversarially crafted DeepFake videos. To address these limitations of Deepfake detectors, we propose FaceSigns, a proactive method for proving media authenticity using semi-fragile neural watermarks. FaceSigns can embed recoverable watermark data into real images and videos at the time of their capture, which can withstand a set of benign image and video transforms while being fragile to malicious tampering such as face swapping.

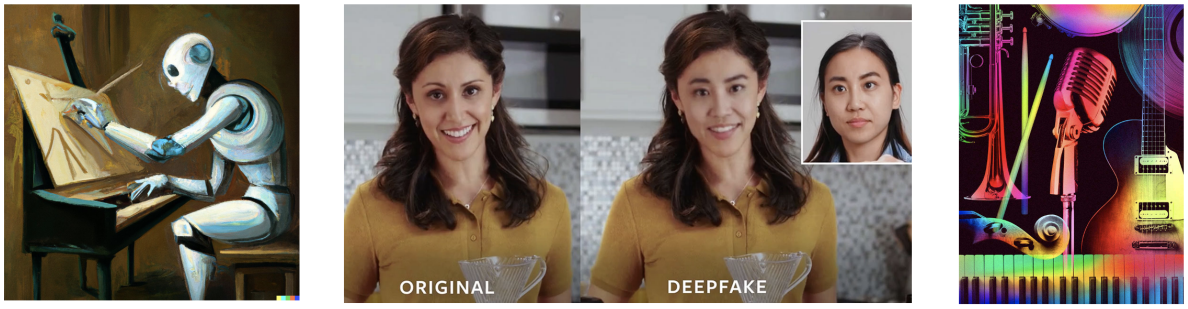
Introduction

My research aims to develop Deep Neural Networks (DNNs) that can synthesize high-quality multimedia content responsibly. The applications of high-quality image, video, and speech synthesis systems are numerous across domains such as entertainment, healthcare, and education. For example, video synthesis systems can be used to create and edit animated films, accompanied by synthesized audio and speech. Additionally, speech synthesis systems can be used for automatic dubbing of movies, creating targeted advertisement campaigns, and generating digital clones of celebrity voices. These systems can also help create interactive educational content. In the healthcare domain, such systems can improve accessibility for visually impaired or speech-impaired individuals.

Media synthesis and manipulation techniques have long existed — Traditionally, heavily engineered computer graphics and signal processing systems were used to synthesize and edit images, videos and audio. DNNs have transformed the domain of media synthesis in two major ways: 1) DNNs have improved the quality of generated content yielding state-of-the-art results for tasks like text-to-speech synthesis, text-to-image synthesis, text-to-video synthesis, image-to-image translation and more. 2) Neural-network-based systems are often trained end-to-end simplifying the synthesis pipeline and making the technology more accessible to people.

As neural networks continue to advance and make high-quality synthetic content more accessible, it is crucial to prioritize safeguards against potential misuse of this technology. The most alarming misuse of this technology is *Deepfakes*. Deepfakes are facially manipulated videos that are used to create a realistic looking scenario that never happened. Such videos can be used to spread misinformation, defame individuals, influence elections or cause political

unrest. Reliable detection or differentiation of real media from synthesized media is important to guard against these potential harms of Deepfake technology. In my research, I investigate the existing methods to detect Deepfake videos. My work unconvers major vulnerabilities in Deepfake detection systems by demonstrating that they can be easily bypassed by an adversary. I then propose a system to for reliable authentication of real media using semi-fragile watermarks.



"A robot painting on a canvas while playing the piano" – by Open AI Dalle2

DeepFake detection challenge by Facebook AI

Open AI Jukebox

Figure 1. Recent examples of DNN-based multimedia generators.

Dissertation Organization

In Chapter 1, I provide a high-level overview of existing neural network based media synthesis techniques techniques. I discuss synthesis techniques both in the vision and audio domain and also discuss the ethical impact of generative media technology.

In Chapter 2, I discuss the systems I developed for natural and expressive speech synthesis. I first describe an expressive voice cloning system that is based on a text-to-speech synthesizer conditioned on speaker and style embeddings. Next, I describe a textless voice conversion system that can convert any given source utterance to a target voice using disentangled representations learnt using self-supervised learning.

In Chapter 3 and Chapter 4 I discuss detection of neural network-generated media. In Chapter 3, I describe the systems that have been developed to detect synthetic media using computer vision classifiers. I then discuss my work, which uncovers major vulnerabilities in such detectors based on adversarial examples. My work demonstrates that state-of-the-art Deepfake

detectors can be easily bypassed using adversarial examples in both white-box and black-box settings.

To address the challenges of synthetic media detectors, In Chapter 4 I describe a semi-fragile media watermarking that we developed to authenticate real images and videos. The watermarking framework embeds a verifiable digital signature into the pixels of the image or video at the time of its capture. The watermark is designed to be semi-fragile to enable fuzzy authentications — The watermark is robust against benign image transforms like compression, and image filters but it is fragile against malicious manipulations such as face replacement.

Chapter 1

Media Synthesis using Neural Networks

Deep Neural Networks (DNNs) have emerged as a powerful tool for synthesizing media, including audio, images, and videos. This has led to a proliferation of AI-generated content, such as images and music generated from text descriptions, facially morphed Deepfake videos and realistic voice clones. While the use of AI-generated media can be beneficial in various applications, such as entertainment, education, and healthcare, it also poses significant risks. The purpose of this chapter is to provide an overview of media synthesis techniques using DNNs, their use cases, and the potential harms associated with them.

DNNs can be used to generate high-quality multimedia content by training synthesis models in both the visual and audio domains. In the vision domain, training techniques like Generative Adversarial Networks (GANs) [56], Variation Autoencoders (VAE) [84] and more recently Stable Diffusion [137] can be used to generate realistic images and videos. These methods are often adapted for tasks such as conditional generation using control variables such as text or labels and face replacement to create a convincing image or footage of events that never occurred.

AI-generated visual content is often accompanied by synthetic audio that is created using DNN-based speech synthesis networks. Text-to-speech synthesis systems have long existed and have become more natural sounding with the advances in neural sequence models such as LSTMs and transformers. However, such systems often lack control over the style aspects of

synthesized speech and are limited to the voices used in the training set. To synthesize convincing speech for synthetic videos, it is important to have fine-grained control over the style aspects of the synthesized speech. Moreover, it is desirable to clone new voices using only a few minutes of data of a given speaker to make neural speech synthesis suitable for practical applications. My research in the domain of speech synthesis addresses the challenges and is described in more detail in Chapter 2.

The ability to generate realistic videos using deep neural networks has many potential use cases. For instance, it can be used for creating realistic training datasets for computer vision applications, such as object recognition and tracking. It can also be used for entertainment, such as creating new films and video games. Additionally, it can be used for virtual and augmented reality applications, allowing users to experience simulated environments and scenarios.

Despite its potential benefits, the use of AI-generated videos also poses significant risks. One of the main risks is the misuse of Deepfake videos, which are videos that are created to misrepresent individuals or events. These can be used to spread false information, create political propaganda, or defame individuals. Additionally, AI-generated videos can be used for malicious purposes, such as cyberbullying, harassment, and blackmail. Furthermore, the ability to generate realistic videos can also have negative impacts on privacy, as individuals can be depicted in situations they never experienced or consented to.

1.1 Visual Synthesis for Facially Manipulated Videos

In this section, I discuss some past works on media synthesis in the vision domain. Image and video synthesis is a broad domain with applications such as unsupervised image generation, text-to-image synthesis, and face replacement and editing. I will focus this section on video synthesis techniques used for synthesizing Deepfake videos, which are facially manipulated videos in which a person‘s face is replaced or modified to simulate a scenario that never occurred. As described in this survey [165], facial manipulations can be classified into three major groups

based on the degree of manipulation shown in Figure 1.1 and described below:

1.1.1 Entire Face Synthesis

This category pertains to methods used for producing complete facial images, typically employing GAN-based techniques, as exemplified by the recent StyleGAN approach proposed in [80]. These methods have been successful in generating high-quality facial images. To generate novel faces, generative models like GANs and VAEs are trained on facial image datasets [104, 24, 20]. Once trained, these models can be used to synthesize new faces by mapping a random noise vector from a uniform distribution to a realistic-looking facial image. These models can also be trained to interpolate between two given images and create interesting intermediate faces of people that do not exist. Recently, several high-quality synthetic facial datasets [1, 38] have been curated using techniques such as ProGAN [79] and StyleGAN [80]. The purpose of these datasets is to train reliable detection methods for synthetic faces.

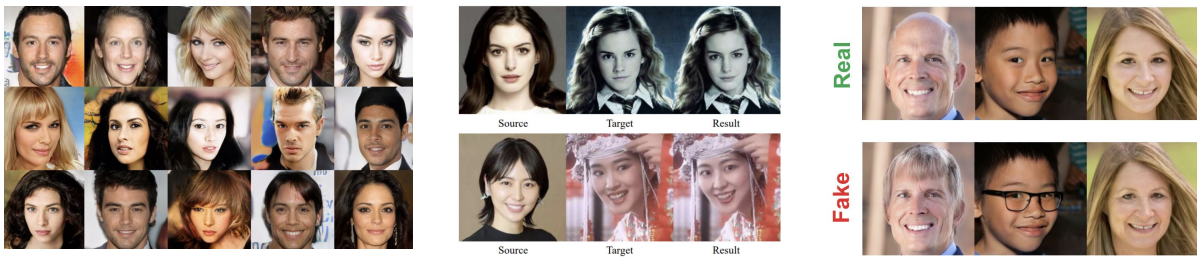


Figure 1.1. Three categories of neural facial manipulation techniques. Entire Face Synthesis includes examples from the website *thispersondoesnotexist.com*. Face Swapping shows examples from SimSwap [26] and attribute manipulation is performed using the FaceApp mobile app.

1.1.2 Face Swapping

Face Swapping methods aim to replace the face of one person in a video with the face of another person. Traditionally, computer graphics-based techniques such as FaceSwap [89] have been used to achieve this goal. In such methods, sparse facial landmarks are detected to extract the face region in an image. These landmarks are then used to fit a 3D template model

which is back-projected onto the target image by minimizing the distance between the projected shape and localized landmarks. Finally, the rendered model is blended with the image and color correction is applied.

More recently neural network-based face swapping methods have been proposed to achieve more realistic-looking synthetic videos. These approaches typically rely on convolutional neural network (CNN) based encoder-decoder networks that can optionally be trained with an adversarial loss. For example, in the DeepFakes method [42] two auto-encoders with a shared encoder are trained to reconstruct the images of source and target face. To create a fake image, the encoded source image is passed as input to the target image decoder. Korshunov and Marcel introduced in [88] the DeepfakeTIMIT database. Fake videos were created using the public GAN-based face-swapping algorithm[1]. In this approach, a CNN based encoder-decoder network is trained with an additional adversarial loss from a discriminator. The encoder-decoder network is trained to reconstruct the original image from a warped image using a reconstruction and an adversarial loss. Similar to [42], during inference a different face can be used as input to get a face-swapped image produced from the encoder-decoder setup.

Face Swapping methods have gained popularity in recent years since they can be easily used to generate Deepfake videos of scenarios that never occurred. The detection of such videos is an important problem and several efforts have been made to identify synthetic videos using machine learning classifiers. I discuss some of these approaches and their limitations in Chapter 3 and Chapter 4.

1.1.3 Attribute Manipulation

Attribute manipulation also known as face editing or face retouching, involves altering various facial features such as the skin or hair color, gender, age, adding glasses, and more [55]. Attribute manipulation is performed by training controllable generative models on facial image datasets. One technique to train controllable generative models is using labelled datasets that

[1] https://github.com/shaoanlu/faceswap-GAN

label the facial attributes in each image. The labels can be used as conditioning variables to train a conditional GAN [112]. For example, in [127], the authors introduced the Invertible Conditional GAN for image editing by jointy training an encoder with a conditional GAN(cGAN) [112].

An alternate approach to achieving attribute control involves learning disentangled latent representations in an unsupervised manner. For example, Lample *et al.* proposed in [93] an encoder-decoder architecture that is trained to reconstruct images by disentangling the salient information of the image and the attribute values directly in the latent space. Methods like [30, 31] learn image-to-image translation across multiple domains without having explicitly labelled paired data between the domains. In this work, the authors trained a conditional attribute transfer network via attribute classification loss and cycle consistency loss.

1.2 Speech Synthesis

In this section, I provide a background of existing work on speech synthesis using neural networks. While there have been some efforts in unsupervised speech synthesis [47], to generate high-quality speech for synthetic videos, the two most common approaches are text-to-speech (TTS) and voice conversion systems. While TTS systems enable speech synthesis from only textual input, they can be limiting in certain use cases that require higher control over the style of the synthesized speech like expressivity, untranscribed sounds, pauses or multi-lingual utterances. Voice conversion systems require a richer input that can be more expensive to collect, but offer fine-grained control over the speaking rate and pitch contour of the synthesized speech.

1.2.1 Text-to-speech (TTS) Synthesis Systems

The best-performing TTS systems decompose the synthesis into two main steps: 1) Synthesizing the perceptually informed spectrogram (mel spectrogram) representation from the text. 2) Vocoding: Synthesizing listenable waveforms from the perceptually informed spectrogram representation. The state-of-the-art TTS methods employ neural networks for both of these steps. Figure 1.2 demonstrates the this two step TTS pipeline.

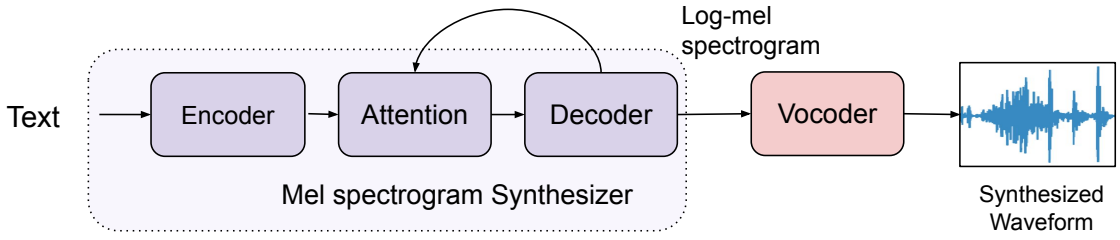


Figure 1.2. Two-step Neural TTS pipeline. A sequence-to-sequence encoder-decoder network first predicts the mel spectrogram from the text sequence. Thereafter, a neural vocoder converts the spectrogram into the waveform representation.

Mel-spectrogram synthesis

Text to mel-spectrogram is typically modelled as a sequence-to-sequence translation problem which is trained on a large corpus of text and speech pairs of a given speaker. Until recently, state-of-the-art TTS models used a Tacotron-based architecture which is an encoder-decoder LSTM model with attention. The model is trained to optimize the reconstruction loss between the generated and ground-truth spectrogram. Once trained, the attention mechanism in the Tacotron model captures the alignment between text tokens and audio frames.

More recently, transformer-based sequence-to-sequence models have shown promising results on TTS benchmarks. The advantage of transformer architecture is that it predicts the spectrogram in parallel as opposed to the auto-regressive nature of LSTMs. To stabilize the training for transformer-based models, intermediate prediction heads for pitch and duration are added that are jointly optimized with the whole network.

While TTS systems can generate natural sounding speech from text, as discussed earlier, they have two main limitations which make it challenging to generate speech for synthetic videos: 1) TTS systems are limited by the speakers used in the training set and cannot directly be used to synthesize speech for new speakers without retraining. 2) TTS models lack control over the style aspects of synthesized speech unless they are trained with additional conditional variables or latent representations. To address the above challenges, I develop an expressive voice cloning framework which is described in Chapter 2.

Vocoder

The need for vocoding arises from the non-invertibility of perceptually-informed spectrograms. These compact representations exclude much of the information in an audio waveform, and thus require a predictive model to fill in the missing information needed to synthesize natural-sounding audio. Notably, standard spectrogram representations discard phase information resulting from the short-time Fourier transform (STFT), and additionally compress the linearly-scaled frequency axis of the STFT magnitude spectrogram into a logarithmically-scaled one. This gives rise to two corresponding vocoding subproblems: the well-known problem of *phase estimation*, and the less-investigated problem of *magnitude estimation*.

Vocoding methodology in state-of-the-art TTS systems [130, 147] endeavors to address the joint of these two subproblems, i.e., to transform perceptually-informed spectrograms directly into waveforms. Wavenet [168] is a high-quality vocoder that uses an autoregressive temporal CNN architecture to vocoder spectrograms into waveforms. However, due to its autoregressive nature, the inference from Wavenet is very slow and not suitable for real-time vocoding. Recently, there have been some works on vocoders that can operate in parallel across the time axis. A recently proposed vocoder, HifiGAN [87] follows a WaveGAN [47] like architecture to vocode spectrograms into waveforms in a parallel manner. Waveflow [133] is another neural vocoder system based on an invertible convolutional architecture that achieves real-time vocoding and is at par with WaveNet in terms of audio quality. In my research [118], we developed a lightweight vocoder that uses a hybrid approach — We use a GAN-based neural network to perform the magnitude estimation and use a heuristic algorithm to perform the phase estimation. For both real spectrograms and synthetic ones from TTS systems, our proposed vocoding method yields significantly higher mean opinion scores than a heuristic baseline and faster speeds than state-of-the-art vocoding methods.

1.2.2 Voice conversion

Voice conversion is the task of changing the voice of a given utterance to a target speaker while preserving the linguistic content. Traditional voice conversion systems rely on parallel speech datasets between two speakers speaking the same sentence [141, 114]. Such systems are typically trained as spectrogram-to-spectrogram translation models using a convolutional encoder-decoder network. However, such systems can only perform voice conversion between the speakers seen during the training, also known as one-to-one or many-to-many voice conversion.

More recently, there have been some developments in voice conversion systems that do not require parallel data [33, 134, 28, 23]. The key idea behind non-parallel voice conversion systems is disentangling speech into representations describing the linguistic content and speaker characteristics. Once the representations are disentangled, a synthesis network is trained to reconstruct the spectrogram from the disentangled representations. Synthesizing speech from these disentangled features allows voice conversion by swapping the speaker embedding of a given utterance with a target speaker.

Previous research on voice conversion has utilized pre-trained automatic speech recognition (ASR) and speaker verification (SV) models to separate content and speaker information from a speech signal [153, 163]. This involves considering the phonetic posteriogram (PPG) or predicted text from the ASR model as the content representation, and the embedding from the SV model as the speaker representation. While this approach has shown potential in voice conversion, it has limitations. Firstly, ASR errors can lead to mispronunciation or inaccurate conversion. Secondly, the content representation (text/PPG) does not capture all linguistic features, resulting in synthesized speech that sounds neutral and lacking in accents, expressions, and speaker-independent style.

1.3 Ethical Impact of Generative Media

Generative AI technologies can be used to create new and original art and design, from paintings and sculptures to clothing and furniture. This can push the boundaries of creativity and lead to new and exciting forms of expression. Deep learning models can generate realistic and immersive virtual worlds for gaming, enhancing the player experience and allowing for greater interactivity and customization. Such technologies can be also used to create personalized learning experiences for students, adapting to their individual learning styles and needs. In medicine, generative models can be used to develop new drugs and treatments.

AI-generated media can amplify existing biases and inequalities in society, such as racial and gender biases, by replicating them in the generated content. For example, a GAN trained on biased data may generate images or text that reinforce harmful stereotypes or discriminatory attitudes. Furthermore, generative models can generate content that may infringe on the intellectual property rights of others, such as copyrights or trademarks. This can create legal challenges for companies and individuals who use these technologies.

Generative models be used to create convincing deepfakes and other forms of manipulated content, which can be used to deceive and defraud individuals and organizations. Fake news generated by AI could be used to sway public opinion and influence democratic processes. One of the most significant challenges is that AI-generated content can be difficult to detect, especially when it is designed to mimic the style and tone of legitimate news sources. In recent years, there have been numerous instances of fake news and misinformation spreading rapidly across social media platforms. These stories can be designed to appeal to people's emotions, playing on fears, prejudices, and biases. When combined with the powerful generative models, fake news can be even more effective at manipulating public opinion. By disseminating false information generated using AI tools and manipulating public opinion, those who control the narrative can influence election outcomes, policy decisions, and public perceptions of important issues.

To address this issue, it is essential to develop effective strategies for detecting and

combating AI-generated media. This can involve a combination of technological solutions, such as AI-powered fact-checking tools, as well as social and political measures, such as public education campaigns and regulations to ensure greater transparency and accountability in online content. Ultimately, it will require a concerted effort from all stakeholders to ensure that the potential harms of AI-generated fake news are minimized and that democratic processes remain robust and resilient. It has become crucial to ensure that these technologies are developed and used responsibly, with a focus on maximizing their benefits while minimizing their potential harms.

Chapter 2

Speech Synthesis for Generative Media

In recent years, there has been a significant advancement in speech synthesis using neural networks. These advancements have enabled high-quality text-to-speech (TTS) synthesis that is almost indistinguishable from natural speech. However, despite these achievements, the speech generated by these systems is often monotonic or neutral in style, lacking the expressiveness and variability that human speech possesses. Additionally, generating unseen voices for new speakers is challenging and requires retraining the models on large amounts of data, making them unsuitable for synthesizing expressive and speaker-adaptive speech that can accompany AI-generated visuals.

To synthesize speech for new speakers using neural networks, past work has focused on two broad problems: *Voice Cloning* and *Voice Conversion*. Voice Cloning is the problem of synthesizing a person's voice from only a few reference audio samples. While voice cloning systems can generate speech from text for a new speaker, it leaves out control over various style aspects of speech. Explicit control over the style aspects of cloned speech is desirable for several applications, such as: voice-overs in animated films, synthesizing realistic and expressive speech for DeepFake videos, translating speech from one language to another while preserving speaking style and speaker identity, advertisement campaigns with expressive speech in multiple voices and languages (etc.).

Voice conversion is the task of modifying an utterance from a source speaker to match the

vocal qualities of the target speaker. Unlike voice cloning systems, there is no text input provided for synthesizing the speech. While traditional voice conversion systems [141, 114] rely on parallel training data with multiple speakers saying the same sentence, there has been a recent surge in voice conversion systems trained on non-parallel multi-speaker datasets [33, 134, 28, 23]. The key idea behind non-parallel voice conversion systems is disentangling speech into representations describing the linguistic content and speaker characteristics. Synthesizing speech from these disentangled features allows voice conversion by swapping the speaker embedding of a given utterance with a target speaker.

In this chapter, I describe two speech synthesis frameworks that I have developed to enable high-quality expressive speech synthesis for new speakers in data-limited settings. The first framework tackles the problem of voice cloning with style control. To achieve style control, we adapt TTS-based voice cloning models by additionally conditioning them on latent and heuristically derived style information. The second framework addresses the challenges in existing voice conversion systems and achieves state-of-the-art results by disentangling content and speaker information features from representations learned using self-supervised learning.

2.1 Speech Synthesis Preliminaries

Generating natural-sounding speech from text is a well-studied problem with numerous potential applications. While past approaches were built on extensive engineering knowledge in the areas of linguistics and speech processing (see [185] for a review), recent approaches adopt neural network strategies which learn from data to map linguistic representations into audio waveforms [4, 54, 130, 173, 147]. Of these recent systems, the best performing [130, 147] are both comprised of two functional mechanisms which (1) map language into *perceptually-informed spectrogram* representations (i.e., time-frequency decompositions of audio with logarithmic scaling of both frequency and amplitude), and (2) *vocode* the resultant spectrograms into listenable waveforms. In such two-step TTS systems, using perceptually-informed spectrograms

as intermediaries is observed to have empirical benefits over using representations which are simpler to convert to audio [130].

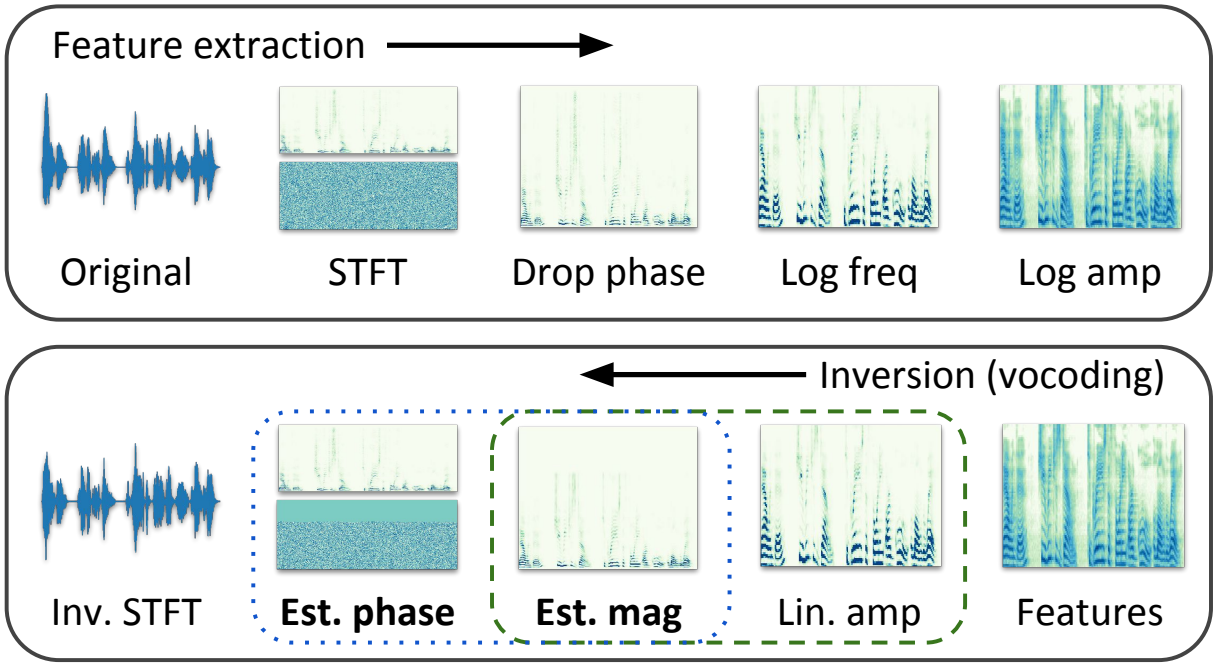


Figure 2.1. Depiction of stages in common audio feature extraction pipelines and corresponding inversion. The two obstacles to vocoding are (1) estimating linear-frequency magnitude spectra from log-frequency mel spectra, and (2) estimating phase information from magnitude spectra.

The typical process of transforming waveforms into perceptually-informed spectrograms involves several cascading stages. Here, we describe spectrogram methodology common to two state-of-the-art TTS systems [130, 147]. A visual representation is shown in Figure 2.1.

Extraction The initial stage consists of decomposing waveforms into time and frequency using the STFT. Then, the phase information is discarded from the complex STFT coefficients leaving only the linear-amplitude magnitude spectrogram. The linearly-spaced frequency bins of the resultant spectrogram are then compressed to fewer bins which are equally-spaced on a logarithmic scale (usually the mel scale [152]). Finally, amplitudes of the resultant spectrogram are made logarithmic to conform to human loudness perception, then optionally clipped and normalized.

Inversion To heuristically invert this procedure (vocode), the inverse of each cascading

step is applied in reverse. First, logarithmic amplitudes are converted to linear ones. Then, an appropriate magnitude spectrogram is estimated from the mel spectrogram. Finally, appropriate phase information is estimated from the magnitude spectrogram, and the inverse STFT is used to render audio.

The audio feature extraction pipeline has two sources of compression: the discarding of phase information and compression of magnitude information. Conventional wisdom suggests that the primary obstacle to inverting such features is phase estimation. However, a systematic evaluation of the individual contributions of magnitude and phase estimation on perceived naturalness of vocoded speech has never been reported.

To perform such an evaluation, in our research, we mix and match methods for estimating both STFT magnitudes and phases from log-amplitude mel spectrograms. A common heuristic for magnitude estimation is to project the mel-scale spectrogram onto the pseudoinverse of the mel basis which was originally used to generate it. As a phase estimation baseline, state-of-the-art TTS research [130, 147] compares to the iterative Griffin-Lim [58] strategy with 60 iterations. We additionally consider the more-recent Local Weighted Sums (LWS) [95] strategy which, on our CPU, is about six times faster than 60 iterations of Griffin-Lim. As a proxy for an ideal solution to either subproblem, we also use magnitude and phase information extracted from real data.

We show human judges the same waveform vocoded by six different magnitude and phase estimation combinations (inducing a comparison) and ask them to rate the naturalness of each on a subjective 1 to 5 scale. Mean opinion scores are shown in Table 2.1.

From these results, we conclude that an ideal solution to *either* magnitude or phase estimation can be coupled with a good heuristic for the other to produce high-quality speech. While the ground truth speech is still significantly more natural than that of ideal+heuristic strategies, the MOS for these methods are only 2-9% worse than the ground truth ($p < 0.05$). Based on these results, we developed a light-weight vocoding method [118] that uses a neural network to solve the magnitude estimation problem, while using the LWS heuristic for phase

Table 2.1. Ablating the effect of heuristics for magnitude and phase estimation on mean opinion score (MOS) of speech naturalness with 95% confidence intervals.

Magnitude est. method	Phase est. method	MOS
Ideal (real magnitudes)	*Ideal* (real phases)	4.30 ± 0.06
Ideal (real magnitudes)	Griffin-Lim w/ 60 iters	3.70 ± 0.07
Ideal (real magnitudes)	Local Weighted Sums	$\mathbf{4.09 \pm 0.06}$
Mel pseudoinverse	*Ideal* (real phases)	$\mathbf{4.04 \pm 0.06}$
Mel pseudoinverse	Griffin-Lim w/ 60 iters	2.48 ± 0.09
Mel pseudoinverse	Local Weighted Sums	2.51 ± 0.09

estimation.

More recently, CNN based parallel vocoders like Waveglow [133] and HiFiGAN [87] have been proposed that solve both magnitude and phase estimation subproblems in an end-to-end manner while staying fast and real-time during inference. We adopt such end-to-end vocoders for developing our voice cloning and voice conversion systems described in the upcoming sections.

2.2 Expressive Neural Voice Cloning

The goal of voice cloning is commonly formulated as learning to synthesize the voice of an unseen speaker using only a few seconds of transcribed or untranscribed speech. This is typically done by embedding speaker-dependent information from the available speech samples of the new speaker, and conditioning a trained multi-speaker Text-to-Speech (TTS) model on the derived speaker embedding [3, 75]. While such a system can achieve promising results in closely retaining speaker-specific characteristics in the cloned speech, it does not offer control over other aspects of speech that are not contained in the text or the speaker-specific embedding. These aspects include variation in tone, speaking rate, emphasis and emotions.

Several past works have focused on the problem of expressive TTS synthesis by learning latent variables for controlling the style aspects of speech synthesized for a given text [174, 149]. Such models are usually trained on a single-speaker expressive speech dataset to learn meaningful latent codes for various style aspects of the speech. Recent works [151, 167], have extended the idea of learning style representations to a multi-speaker setting by conditioning the TTS

synthesis model on both speaker identity and style encodings. Such techniques show promise in disentangling style and speaker specific information, and generate different style variants of synthesized speech for the same text and speaker. However, these methods are limited by the speakers used in the training set and cannot be directly used for synthesizing voices of speakers not seen during training.

Adapting multi-speaker TTS models for voice cloning requires scaling up model training to a large multi-speaker TTS dataset, containing several minutes of transcribed speech from thousands of speakers. High speaker diversity in the training data is important to achieve generalization on unseen speakers [3, 75]. The goal of our voice conversion framework is to perform TTS synthesis for an unseen speaker with control over the style aspects of generated speech. As a first step in this direction, we train a TTS model conditioned on speaker encodings and latent style tokens [174] on a large multi-speaker dataset. While this model is able to generate voices for unseen speakers, we find that the results fall short in terms of speech naturalness and style control during synthesis. Our results suggest that learning meaningful latent style aspects is difficult when training on a large multi-speaker dataset containing speech with mostly neutral style and expressions.

To address problem of disentangling style and speaker characteristics on a large multi-speaker dataset containing mostly style-neutral speech, we propose a voice cloning model that is conditioned on both latent and heuristically derived style information. Specifically, we condition our TTS synthesis model on (i) text, (ii) speaker encoding (iii) pitch contour of the target speech and (iv) latent style tokens [174]. By conditioning synthesis on various style aspects and speaker embeddings derived from the target speech, we are able to train a model that offers fine-grained style control for synthesized speech. To adapt inference for an unseen speaker, we can either perform zero-shot inference or fine-tune the synthesis model on the limited text and speech pairs for the new speaker. Through both quantitative and qualitative evaluations, we demonstrate that our proposed model can make a new voice express, emote, sing or copy the style of a given reference speech.

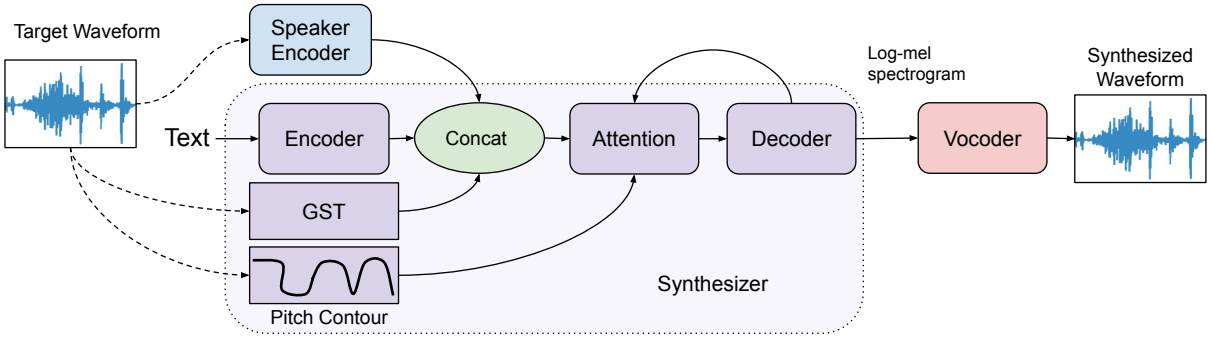


Figure 2.2. Expressive Voice Cloning Model: Tacotron-2 TTS model conditioned on speaker and style characteristics derived from the target audio of a given text. At inference time, the model can be provided independent references for style and speaker encodings to achieve expressive voice cloning.

2.2.1 Voice Cloning Framework

Our expressive voice cloning framework is a multi-speaker TTS model that is conditioned on speaker encodings and style aspects of speech. Style conditioning in expressive TTS models is popularly done by learning a dictionary of latent style vectors called Global Style Tokens (GST) [174]. While GSTs can learn meaningful latent codes when trained on a dataset with high variation in expressions, we empirically find that it offers limited style control when trained on a large multi-speaker dataset with mostly neutral prosody.

Signal processing heuristics like the Yin algorithm [41] can derive the fundamental frequency contour (pitch contour) and voicing decisions from speech, which can be useful for expressive speech synthesis. We find that using a combination of latent and heuristically derived style information in the TTS model not only provides fine-grained control over the style aspects of synthesized speech, but also scales up to a large multi-speaker dataset to produce more natural sounding audio for an unseen speaker.

Speaker Encoder

Speaker conditioning in multi-speaker TTS models is usually done using a lookup in the speaker embedding matrix which is randomly initialized and trained end-to-end with the synthesizer. While such a framework learns speaker-specific information via the embedding

vectors, synthesis cannot be generalized to unseen speakers. To adapt the multi-speaker TTS model for the goal of voice cloning, the speaker embedding layer can be replaced with a speaker encoder that derives speaker specific information from the target waveform. In this setting, the speaker encoder can obtain embeddings for speakers not seen during training using a few reference speech samples. To obtain meaningful embeddings, the speaker encoder should be trained to discriminate between different speakers for the task of speaker verification [171]. We follow the speaker encoder architecture described in [171, 108]. The speaker encoder is trained to optimize a generalized end-to-end speaker verification loss [171], that encourages high cosine similarity between embeddings from same speaker and low similarity between different speaker embeddings. During inference, each utterance is broken into smaller segments of 1,600 ms with 1,000 ms overlap between consecutive segments. The final embedding is estimated by averaging the embedding of each individual segment.

Mel-Spectrogram Synthesizer

The goal of our synthesis model is to disentangle the style and speaker-specific information in speech by conditioning our TTS synthesis model on the speaker encoding and various style aspects. To this end, we adapt the synthesis model used in Mellotron [167] for the task of voice cloning. Mellotron is a multi-speaker TTS model that extends Tacotron 2 GST [174] by additional conditioning on pitch contours and speaker embeddings. To adapt Mellotron for voice cloning, we remove the speaker embedding layer and replace it with the speaker encoder network described in Section 2.2.1.

At its core, our synthesis model based on Tacotron 2 [147], is an LSTM based sequence-to-sequence model composed of an encoder that operates on a sequence of characters and a decoder that generates the individual frames of the mel spectrogram while attending over the encoded representations. Along with the encoded representation for text, we concatenate the speaker encoding (obtained from the speaker encoder) and the GST embedding at each time-step. The GST embedding is obtained by querying a dictionary of latent style vectors with the target

mel-spectrogram using a multi-headed attention mechanism described in [174]. Decoding occurs in an autoregressive manner where we synthesize one mel spectrogram frame at a time by providing the fundamental frequency (from the pitch contour) and the mel spectrogram of the previous frame as the input to the decoder. The pitch contours are derived from the target speech using the Yin algorithm with harmonicity thresholds between 0.1 and 0.25.

In this way, we can factor mel-spectrogram synthesis into the following variables: *text (t)*, *speaker encoding (s)*, *pitch contour (f_0)* and *latent style embedding obtained from GST (z)*. Formally, our synthesizer is a generative model $g(t, s, f_0, z; W)$ that is parameterized by trainable weights W, trained to optimize a loss function L that penalizes the differences between the generated and ground truth mel spectrogram. That is,

$$\min_{W} \mathbb{E}_{(t_i,a_i)\sim D} \{L(g(t_i, s_i, f_{0i}, z_i; W), mel_i)\} \tag{2.1}$$

where D is the dataset containing text and audio pairs (t_i, a_i). The variables $(s_i, f_{0i}, z_i, mel_i)$ are all derived from the target waveform a_i. For the loss function L, we use the L2 loss between the generated and ground truth mel spectrograms.

During training, the synthesizer learns another latent variable: the attention map between the encoder and decoder states which captures the alignment between text and audio. Following the notation used in [167], we call this latent variable *rhythm*, since it controls the timing aspects of synthesized speech. Note that unlike other style aspects which can be obtained directly from a_i, deriving *rhythm* requires both text and audio (t_i, a_i). In our experiments, we obtain the *rhythm* by using our synthesizer as a forced-aligner. That is, for a given text and audio pair, we derive the attention map between the encoder and decoder states by doing a forward pass through our model using teacher forcing. Therefore, during inference, our synthesizer g can be explicitly conditioned on rhythm r derived from some text and audio pair: $g(t, s, f_0, z, \boldsymbol{r}; W)$.

While the style aspects are obtained from the target waveform of the same speaker during training, we can use a different reference audio and text pair during inference. For example, we

can transfer the pitch contour and rhythm of a style reference audio S from a different speaker to the voice of a given target speaker T as follows:

$$mel = g(t_S, s_T, f_{0S}, z_T, r_S; W) \tag{2.2}$$

The output *mel* should have the same pitch and rhythm as the style reference S and should retain the latent style aspects and voice of the target speaker T.

Additionally, to assess the importance of pitch contours during training, we train another TTS model that is conditioned only on the latent style aspects obtained using GST. We use the same Tacotron2 architecture and GST module as our proposed model. Formally, this alternative synthesizer $g(t, s, z; W)$ is trained to optimize the same objective as Equation 2.1:

$$\min_{W} \mathbb{E}_{(t_i, a_i) \sim D} \{L(g(t_i, s_i, z_i; W), mel_i)\} \tag{2.3}$$

We refer to this alternative model as *Tacotron2 + GST* in our experiments. Similar to our proposed system, this model can also be additionally conditioned on rhythm. Since we are not explicitly conditioning the model on pitch contours, we expect the pitch variation in speech to be captured as part of the latent style tokens. We empirically demonstrate that using only latent style representation on a large multi-speaker dataset with neutral prosody offers limited style control and audio naturalness.

Vocoder:

For decoding the synthesized mel-spectrograms into listenable waveforms, we use the WaveGlow [133] model trained on the single speaker Sally dataset [167]. An advantage of WaveGlow over WaveNet [168] is that it allows real-time inference, while being competitive in terms of audio naturalness. The same vocoder model is used across all experiments and datasets. We find that the vocoder model trained on a single speaker generalizes well across all speakers in our datasets.

2.2.2 Cloning Techniques: Zero-Shot and Model Adaptation

We adopt the following two approaches for cloning the voice of a new speaker from a few transcribed or untranscribed speech samples:

Zero-Shot: For zero-shot voice cloning, we derive the speaker embedding by taking the mean followed by L-2 normalization of the speaker encodings of the individual samples of the target speaker. Since speaker encodings are obtained directly from the waveforms, we do not require audio transcriptions of the new speaker for zero-shot voice cloning.

Model Adaptation: When transcribed samples of a new speaker are available, we can fine-tune our synthesis model using the text and audio pairs. As shown in Neural Voice Cloning [3], fine-tuning can significantly improve the speaker similarity metrics of the cloned speech. Also, the authors of [3] observe that fine-tuning the whole synthesis model is faster and more effective than fine-tuning only the speaker embedding layer since more degrees of freedom are allowed in the whole model adaptation. Our preliminary experiments on model adaptation suggested the same. We hypothesize the reason for this is that fine-tuning the last-few layers of the synthesis model is essential, if not sufficient, to adapt the synthesizer to the speaker-specific speech characteristics. Therefore, we study the following two model adaptation techniques: **Adaptation whole -** Fine-tune all the parameters of the synthesis model on the text and audio pairs of the new speaker. **Adaptation decoder -** Fine-tune only the decoder parameters of the synthesis model. The advantage of only adapting the decoder parameters is that it requires fewer speaker-specific model parameters and a shared encoder can be used across all speakers in a real-world deployment setting. In both of the above adaptation settings, we fine-tune our model for 100 to 200 iterations using Adam optimizer with a learning rate of $1e-4$.

2.2.3 Experiments on Expressive Voice Cloning

We train our mel-spectrogram synthesis model on the clean subset of the publicly available Libri-TTS [184] dataset—*train-clean-100* and *train-clean-360*. This clean subset

contains around 245 hours of speech across 1151 speakers sampled at 24 kHz. We filter out utterances longer than 10 seconds and resample waveforms to 22050 Hz. The speaker embedding layer is replaced with our speaker encoding network which is kept frozen during training. We use a validation set with 250 examples and train the model using a batch size of 32 and an initial learning rate of 5e-4. We use an Adam optimizer [83] to update the weights and anneal the learning rate to half its value every 50k mini-batch iterations. For the *Tacotron 2 + GST* model, we use the same Tacotron 2 architecture and GST hyper-parameters as our proposed model. Training for the proposed model and the *Tacotron 2 + GST* model converged in 210,000 and 185,000 mini-batch iterations respectively and took around 4 seconds per iteration on a single Nvidia Titan 1080 GPU. The Resemblyzer speaker encoder [108, 107] used in our experiments is trained on the VoxCeleb [116], VoxCeleb2 [35] and LibriSpeech-other [123] datasets containing a total of 8.4k speakers. The authors of [107] report a 4.5% Equal Error Rate (EER) for the task of speaker verification using this speaker encoder on their internal test set.

To evaluate our cloning techniques objectively in terms of style and speaker disentanglement, and also assess their usefulness in real world settings, we perform the following cloning tasks:

1. Text *Cloning speech directly from text:* For cloning speech directly from text, we first synthesize speech for the given text using a single speaker TTS model: Tacotron 2 + WaveGlow trained on the LJ Speech [74] dataset. We then derive the pitch contour of the synthetic speech using the Yin algorithm [41] and scale the pitch contour linearly to have the same mean pitch as that of the *target speaker samples*. For deriving rhythm, we use our proposed synthesis model as a forced aligner between the text and Tacotron2-synthesized speech. We use the *target speaker samples* for obtaining the GST embedding for both our proposed model and the baseline Tacotron2 + GST model.

2. Imitation - *Reconstruct a sample from the target speaker:* In this setup, we use a text and audio pair of the target speaker (not contained in the *target speaker samples*), and try to reconstruct the audio from its factorized representation using our synthesis model. All of the

style conditioning variables - pitch, rhythm and GST embedding are derived from the speech sample we are trying to imitate. The imitation task is a toy experiment that allows quantitative evaluation of style similarity metrics between the synthesized speech and style reference.

3. Style Transfer - *Transfer the pitch and rhythm of speech from a different expressive speaker:* The goal of this task is to transfer the pitch and rhythm from some expressive speech to the cloned speech for the target speaker. For this task, we use examples from the single speaker Blizzard 2013 dataset [82] as style references. This dataset contains expressive audio book readings from a single speaker with high variation in emotion and pitch. For our proposed model, we use this *style reference audio* to extract the pitch and rhythm. Similar to the Text task, we scale the pitch contour to have the same mean as that of the *target speaker samples*. In-order to retain speaker-specific latent style aspects, we use *target speaker samples* to extract the GST embedding. For the Tacotron2 + GST model, which does not have explicit pitch conditioning, we use the *style reference audio* for obtaining the GST embedding and the rhythm.

For the above-described cloning tasks, we evaluate three aspects of the cloned speech: i) speaker similarity to the target speaker, ii) style similarity to the reference style and iii) speech naturalness. We encourage the readers to listen to our audio examples referenced in the footnote of the first page to contextualize the following results.

Speaker Classification Accuracy: We train a speaker classifier on the VCTK dataset to classify a given utterance as one of the 108 speakers. The speaker classifier is a two layer neural network with 256 hidden units that takes as input the speaker encoding obtained through our pre-trained speaker encoder network. Similar to [3], our speaker classifier achieves 100% accuracy on a hold out set containing 200 examples from the VCTK dataset. However, since our classification model and training dataset for the synthesizer are not the same as [3] (1,151 speakers in ours vs. 2,481 speakers in [3]), we do not make direct comparisons with their work.

We conduct our speaker classification evaluations on all 108 speakers of the VCTK dataset. We clone 25 speech samples per speaker for each cloning task. Figure 2.3 (left) shows the speaker classification accuracy curves for all cloning tasks and techniques with respect to

the number of target speaker samples. Our results are consistent with the following findings of [3]—Model adaptation significantly outperforms the zero-shot voice cloning technique since it allows the model to adjust to the speaker characteristics of the new speaker. More target speaker samples helps improve speaker classification accuracy, although in the zero-shot scenario we do not observe much improvement after 10 target speaker samples.

For zero-shot voice cloning, both Tacotron2-GST and our proposed model achieve similar speaker classification accuracy for *Text* and *Style Transfer* cloning tasks. The accuracy of our proposed model is slightly higher for the imitation task as compared to other tasks for both model adaptation and zero-shot voice cloning. This implies that conditioning on the actual pitch contour of the target speaker improves speaker specific characteristics of the cloned speech. While linear scaling of a reference style pitch contour works well, our findings motivate future research on predicting speaker-specific pitch contours from text and speaker encodings.

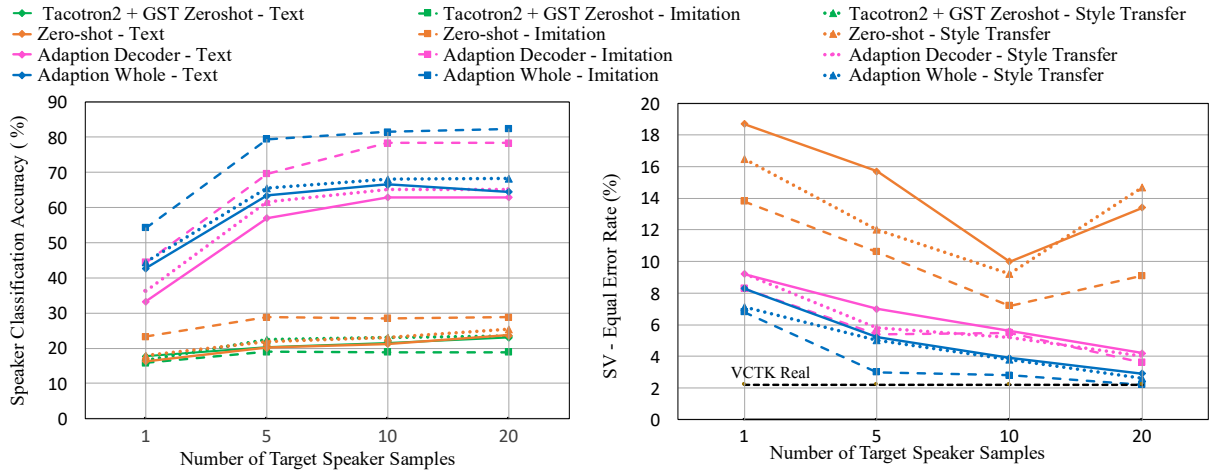


Figure 2.3. Speaker similarity evaluation of each cloning technique for different voice cloning tasks in terms of Speaker Classification Accuracy and Speaker Verification Equal Error Rate (SV-EER).

Speaker verification Equal Error Rate (SV-EER): SV-EER is another objective metric used to evaluate speaker similarity between the cloned audio and the ground-truth reference audio. We use a speaker verification system that scores the speaker similarity between two utterances based on the cosine similarity of the encodings obtained using the speaker encoder described

in Section 2.2.1. Equal Error Rate (EER) is the point when the false acceptance rate and false rejection rate of the speaker verification system are equal.

We perform speaker verification evaluations on randomly selected 20 speakers in the VCTK dataset. We enroll 5 speech samples per speaker in the speaker verification system and synthesize 50 speech samples per speaker for each cloning task. EERs are estimated by pairing each sample of the same speaker with another sample from a different speaker. Figure 2.3 shows the plots of SV-EER for different cloning techniques and tasks using our proposed model and also the those estimated using real data. Our observations on the SV-EER metric are similar to those on the accuracy metric. Model adaptation outperforms zero-shot cloning techniques and with more than 10 cloning samples achieves similar EER as the real data.

Table 2.2. Style similarity evaluations for the imitation and style transfer tasks. We use three objective error metrics (lower values are better). For the style transfer task we present the mean opinion scores on style similarity (Style-MOS) with 95% confidence interval.

	Imitation			*Style Transfer*
Approach	GPE	VDE	FFE	Style-MOS
Tacotron2 + GST - Zero-shot	20.37%	26.39%	29.47%	2.69 ± 0.11
Proposed Model - Zero-shot	3.72%	10.65%	11.74%	3.15 ± 0.11
Proposed Model - Adaptation Whole	2.97%	12.58%	13.60%	3.40 ± 0.10
Proposed Model - Adaptation Decoder	2.39%	11.60%	12.51%	3.29 ± 0.10

Style Similarity: In order to evaluate the similarity between the style of synthesized and reference audio, we perform quantitative evaluation on the Imitation task. We use the following error metrics: Gross Pitch Error (GPE) [117], Voicing Decision Error (VDE) [117] and F0 Frame Error (FFE) [34]. Results are presented in Table 2.2 in which we compare the error values for different approaches when using 10 target speaker samples for cloning. We synthesize 25 speech samples per speaker for all speakers in the VCTK dataset to estimate the reported error values. Our proposed models significantly outperform the Tacotron 2 + GST baseline, clearly indicating the importance of pitch contour conditioning for accurate style transfer.

We also conduct a crowd-sourced listening test on Amazon Mechanical Turk (AMT) for

the style transfer task in which we ask the listeners to rate the style similarity between the ground truth style reference and synthesized audio on a 5 point scale. For each cloning technique (using 10 target speaker samples), we synthesize 25 audio samples per speaker for 20 speakers in the VCTK dataset leading to 500 evaluations of each technique. We present the style similarity Mean Opinion Scores (Style-MOS) in Table 2.2. It can be seen that our proposed models significantly outperform the Tacotron 2 + GST model. Model adaptation techniques perform slightly better than zero-shot method suggesting that fine-tuning improves the model predictions for an unseen speaker encoding.

Naturalness: To assess speech naturalness, we conducted a crowd-sourced listening test on AMT and asked listeners to rate each audio utterance on a 5-point naturalness scale to collect Mean Opinion Scores (MOS). Similar to the above mentioned user study, we use 10 target speaker samples for each cloning technique. All evaluations are conducted on randomly selected 20 VCTK speakers with 25 audio samples synthesized per speaker. Each sample is rated independently by a single listener leading to 500 evaluations for each technique per cloning task. We report the MOS of Real data and audio synthesized using different cloning techniques in Table 2.3. Our proposed model significantly outperforms the baseline Tacotron2 + GST model for both zero-shot and model adaptation techniques. This suggests that pitch contour conditioning in a multi-speaker setting helps improve speech naturalness in addition to providing higher style similarity. It can be seen that the naturalness is even further improved with model adaptation techniques since it allows the generative model to adjust for the unseen speaker encodings.

2.3 Voice Conversion Using Iterative Self-Refinement

In this section, I describe a zero-shot voice conversion framework we developed using speech representations trained with self-supervised learning. The key idea behind our voice conversion system is disentangling speech into features describing the linguistic content and speaker characteristics. Synthesizing speech from these disentangled features allows voice

Table 2.3. Mean Opinion Score (MOS) for speech naturalness with 95% confidence intervals.

	Cloning Task		
Approach	Text	Imitation	Style Transfer
Real data VCTK		4.11 ± 0.08	
Real data Blizzard		4.07 ± 0.08	
Tacotron2 + GST - Zero-shot	2.67 ± 0.10	2.51 ± 0.10	3.02 ± 0.09
Proposed Model - Zero-shot	3.56 ± 0.09	3.54 ± 0.10	3.53 ± 0.10
Proposed Model - Adaptation Whole	3.75 ± 0.09	3.71 ± 0.09	3.60 ± 0.09
Proposed Model - Adaptation Decoder	3.61 ± 0.09	3.57 ± 0.09	3.45 ± 0.09

conversion by swapping the speaker embedding of a given utterance with a target speaker.

To derive disentangled speech representations in a text-free manner, recent methods [92, 131, 101, 68, 28] have proposed to obtain speaker information from a speaker verification model and linguistic content information from the output of models trained using self-supervised learning (SSL) [7, 62]. While the representations obtained from the SSL models are highly correlated with phonetic information, they also contain speaker information [71, 68, 70]. To remove speaker information from the SSL model outputs, some techniques utilize an information bottleneck approach such as quantization [131, 92, 60]. Alternatively, several researchers have proposed training strategies that employ an information perturbation technique to eliminate speaker information without quantization [135, 28, 29, 71]. Notably, for training synthesizers, NANSY [28] and NANSY++ [29] propose to heuristically perturb the voice of a given utterance with hand-engineered data augmentations, before deriving the content embedding from the SSL model. To reconstruct the original audio accurately, the synthesizer is forced to derive the speaker characteristics from the speaker embedding since the speaker information in the content embedding is perturbed. While such techniques are effective, heuristic voice perturbation algorithms based on pitch randomization and formant shifting represent a very limited set of transformations. We hypothesize that such training strategies can be improved by utilizing neural network-generated augmentations.

In our work, I propose a learning framework to automatically generate diverse data

transformations during training and enable controllable speech synthesis from imperfectly disentangled but uncompressed speech representations. First, we develop a feature extraction methodology that not only derives the content and speaker embeddings but also prosodic information such as speaking rate and pitch modulation. Next, to train a controllable synthesizer, we propose a training strategy that utilizes the synthesis model itself to create challenging voice-converted transformations of a given speech utterance. At any given training iteration, the current state of the synthesis model is used to transform the input content embedding and the model is updated to minimize the reconstruction error of the original utterance.

All the components in our framework are trained in a text-free manner requiring only audio data. Once trained, our framework can be used for tasks such as zero-shot voice conversion, audio reconstruction with pitch and duration modulation as well as multilingual voice conversion across languages outside of the training set. On metrics evaluating speaker similarity, intelligibility and naturalness of synthesized speech we demonstrate that our model outperforms previously proposed zero-shot voice conversion methods for both seen and unseen speakers.

2.3.1 Related Work

Voice conversion: Voice conversion is the task of modifying an utterance of a source speaker to match the vocal qualities of a target speaker. Traditionally, voice conversion models were trained as a speech-to-speech translation system on a parallel dataset containing multiple speakers saying the same utterance [154, 25]. More recently, voice conversion systems have been developed by training neural synthesizers to reconstruct speech from disentangled representations describing content and speaker characteristics [134, 33]. For example, [153, 163] have utilized pre-trained automatic speech recognition (ASR) and speaker verification (SV) models to disentangle content and speaker information respectively. The predicted text or phonetic posteriogram (PPG) obtained from the ASR model is taken as the content representation. However, such voice conversion systems have limitations: 1) Training such systems requires transcribed speech data and the synthesis is limited to the language the model is trained on. 2) Text and PPG do not

capture all linguistic features such as accent, expressions, emotions or speaker-independent style resulting in neutral-sounding synthesized speech.

To derive linguistic content in a text-free manner, some prior works have utilized SSL based models. However, as noted by prior work [131, 68], SSL model outputs do not necessarily separate speaker and content information. One line of research [131, 92, 60] aiming to disentangle the speaker and content representations, proposes an information bottleneck approach to quantize SSL model outputs thereby limiting the information to only capture the content or pseudo-text of the audio. However, the loss of information during such a quantization approach leads to sub-optimal reconstruction quality.

Addressing the limitations of information bottleneck approaches, researchers have proposed training strategies based on heuristic transformations. For example, in ContentVec [135] and ACE-VC [71], while training the SSL-based feature extractor model, the audio is transformed using pitch-shift transformation and the SSL model is encouraged to output similar representations for the original and transformed audio. Alternatively, in NANSY [28], the transformations are applied while training the synthesizer, i.e. the synthesizer is tasked to reconstruct the original audio from the content representation of audio perturbed using transforms such as formant-shift, pitch-randomization and randomized frequency shaping. Although these heuristic transformations serve as a reasonable proxy for voice conversion methods, we hypothesize such methods can be greatly improved by utilizing the voice conversion system itself to generate more diverse input transformations.

Transformation invariant representation learning: Learning representations that are invariant to various input transformations has been a topic of significant interest in unsupervised representation learning [6, 113]. Several techniques addressing this challenge utilize domain-specific and hand-engineered data augmentation methods [27, 22, 164, 59, 113] for training transformation invariant representation encoders. Stochastic data augmentation in the image domain such as cropping, rescaling, shifts in brightness and recoloring have been popularly used [27, 164] to learn robust representations for image classification tasks. More recently, [158]

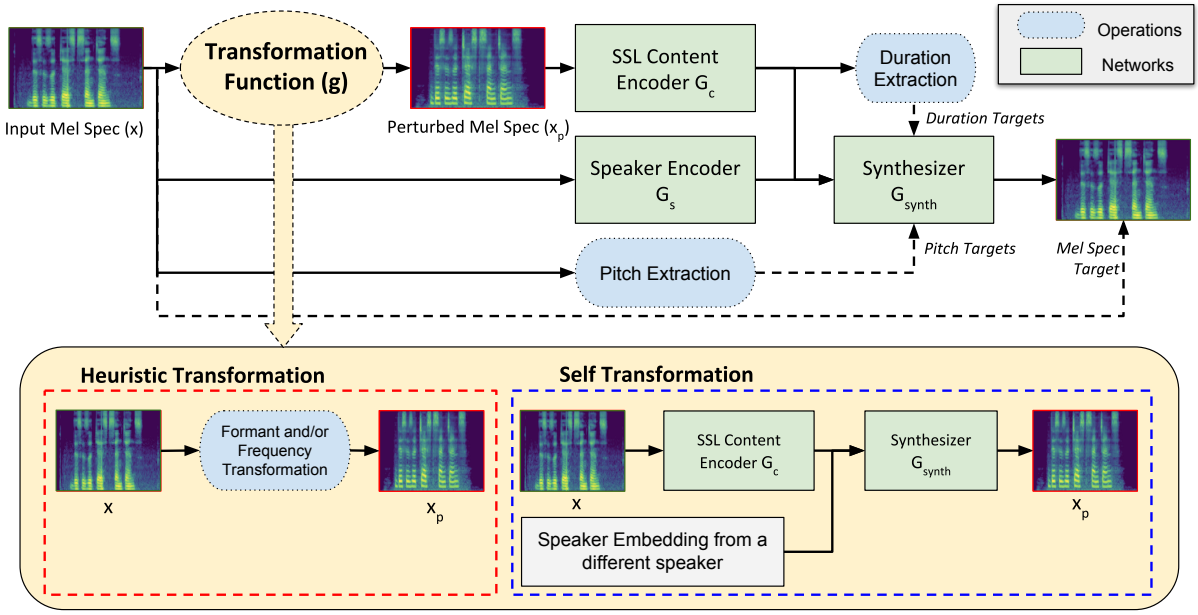


Figure 2.4. Voice Conversion Approach Overview: The synthesis model is trained to reconstruct the mel-spectrogram from SSL-based content representation of a transformed audio (heuristic or self-transformed) and speaker embedding of the original audio.

proposed to train generative models to produce diverse views from a given input by adding a bounded perturbation. Their results demonstrate that neural generative models can produce a more diverse set of input distortions (compared to hand-engineered augmentations) without requiring domain-specific knowledge. While such techniques have been useful for learning transformation invariant representations for downstream recognition tasks, their applicability for upstream generative tasks is yet to be explored. In our work, we develop a novel framework for training a controllable synthesis model using self-generated input transformations. In contrast to previous ideas, we do not introduce additional networks for data augmentation but utilize the synthesizer model itself to generate diverse input transformations.

2.3.2 Voice Conversion Approach

Our framework consists of two main components: 1) A feature extractor that derives content (linguistic features), speaker and style representations from a given speech utterance. 2) A synthesizer that reconstructs the audio from the derived representations. To allow controllable

synthesis from imperfectly disentangled representations, we propose a training strategy that challenges the model to reconstruct the audio from self-generated perturbations of the content representation. Specifically, we train the model to reconstruct the audio from the content representation of a heuristically modified or self transformed audio, while preserving the speaker and style representations. The content and speaker encoder networks remain fixed during synthesis model training.

Feature Extraction

Content Embedding: We define content as a temporal feature that encodes the linguistic information of a given speech utterance. We use the output of the Conformer-SSL [62] model (G_c) as the content representation of speech (z). The Conformer-SSL model is a convolution-augmented transformer architecture that is trained to reconstruct the masked areas of the mel-spectrogram on $56k$ hours of English speech data, using contrastive and masked language modelling (MLM) losses. The representations derived from SSL-based speech encoder models have been shown to have a high correlation with corresponding phonetic information [7]. Given a speech utterance as a sequence of mel-spectrogram frames $x = x_1 \dots x_T$, the Conformer-SSL model outputs a temporally downsampled sequence of feature vectors $z = G_c(x) = z_1 \dots z_{T'}$. In our setup, the SSL model temporally downsamples the mel-spectrogram by a factor of 4 and the output at each time-step z_t is a 256 dimensional vector corresponding to a contextualized representation of roughly 46 milliseconds of audio.

Speaking Rate or Duration: Speaking rate determines how long the speaker vocalizes each phoneme of a given utterance. Since the speaking rate can vary greatly across different speakers and accents, accurate modelling of speaking rate during synthesis is important to closely mimic a target speaker. We propose a technique to derive the speaking rate or duration information in a text-free manner from the content representation. Since SSL representations have a high correlation with phonemes [7, 62], we conjecture that if a phoneme is emphasized in an utterance, the consecutive content vectors at the corresponding timesteps will have high similarity.

Therefore, we propose Algorithm 1 to process the content representation $z = z_1 \dots z_{T'}$ into a duration-augmented content representation $z' = z'_1 \dots z'_{T'}$ and $d' = d'_1 \dots d'_{T'}$. We group together consecutive content vectors with cosine similarity higher than a threshold τ, and set the target duration for the averaged vector as the number of grouped vectors times the duration of a single vector.

Algorithm 1. Deriving duration-augmented content by grouping similar consecutive vectors

$z' \leftarrow [z_1]$ ▷ Initialize z' with the vector from the first time-step in z
$d' \leftarrow [\delta]$ ▷ d'_t represents duration of z'_t. δ represents duration of of each z_t (i.e 46 ms)
$num\_grouped \leftarrow 1$ ▷ number of similar vectors grouped at the last processed time-step
for $t \leftarrow 2$ **to** T' **do**
  if $CosineSimilarity(z_t, z'[-1]) > \tau$ **then** ▷ Group z_t with the running group
    $z'[-1] \leftarrow (z_t + num\_grouped * z'[-1])/(num\_grouped + 1)$ ▷ Update average
    $d'[-1] \leftarrow \delta * (num\_grouped + 1)$
    $num\_grouped \leftarrow num\_grouped + 1$
  else ▷ Insert z_t in a new group
    $z'.append(z_t)$
    $d'.append(\delta)$
    $num\_grouped \leftarrow 1$
  end if
end for
return z', d'

Speaker Embedding: The speaker embeddings in our setup are derived from the TitaNet [85] model (G_s). TitaNet is based on a 1-D depthwise separable convolution architecture with Squeeze and Excitation layers that provide global context. The TitaNet speaker verification model is trained using additive angular margin loss [102] on 3373 hours of speech from multiple datasets that span 16681 speakers. The model is designed to be parameter-efficient and achieves state-of-the-art results on the VoxCeleb-1 speaker verification benchmark with an EER of 0.68%. The output from the speaker verification model is a 256 dimensional speaker embedding $s = G_s(x)$.

Pitch Contour: The pitch contour p is derived from the fundamental frequency f_0 contour of the speech signal that represents the prosodic modulations over time. The raw values in the fundamental frequency contour (derived from PYin algorithm [110]) are speaker-dependent, therefore f_0 is not strictly disentangled from the speaker information. To ensure that the pitch

contour only encodes the prosodic changes and not the speaker identity, we normalize f_0 using the mean (f_{mean}) and standard deviation (f_{std}) of all pitch contours of the given speaker. That is, $p = (f_0 - f_{mean})/f_{std}$.

Synthesizer

The task of the synthesizer is to first reconstruct the ground-truth mel-spectrogram from the extracted speech representations and then vocode the mel-spectrogram into a listenable audio waveform. For vocoding, we use a HiFiGAN [87] vocoder, which is trained separately on spectrogram and waveform pairs of real audio from a multi-speaker dataset.

Our mel-spectrogram synthesizer G_{synth} is composed of two feed-forward transformers F_e and F_d and intermediate modules to predict the duration and pitch similar to [94] but operates on the grouped content representation $z' = z'_1 \dots z'_{T'}$ instead of text. The speaker embedding s is repeated across all time-steps and concatenated with each z'_t to be fed as input to the first feed-forward transformer F_e. The hidden representation from F_e is then used to predict the duration and pitch, that is: $h = F_e(z', s)$; $\hat{y_d} = DurationPredictor(h)$, $\hat{y_p} = PitchPredictor(h)$. The pitch contour is projected and averaged over each time-step of the hidden representation h and added to h to get $k = h + PitchEmbedding(p)$. Finally, k is discretely upsampled as per the ground-truth duration d' and fed as input to the second transformer F_d to get the predicted mel-spectrogram $\hat{y} = F_d(DurationRegulation(k, d'))$

Our model is trained to optimize three losses — mel-reconstruction error, pitch prediction error and duration prediction error such that

$$L_{synth} = \|\hat{y} - y\|_2^2 + \lambda_1 \|\hat{y_p} - p\|_2^2 + \lambda_2 \|\hat{y_d} - d'\|_2^2 \tag{2.4}$$

During inference, we can use either the predicted pitch and duration, in which case the prosody is derived from both the content and speaker embeddings; or we can mimic the prosody and speaking rate of the source utterance by using ground-truth duration and pitch information.

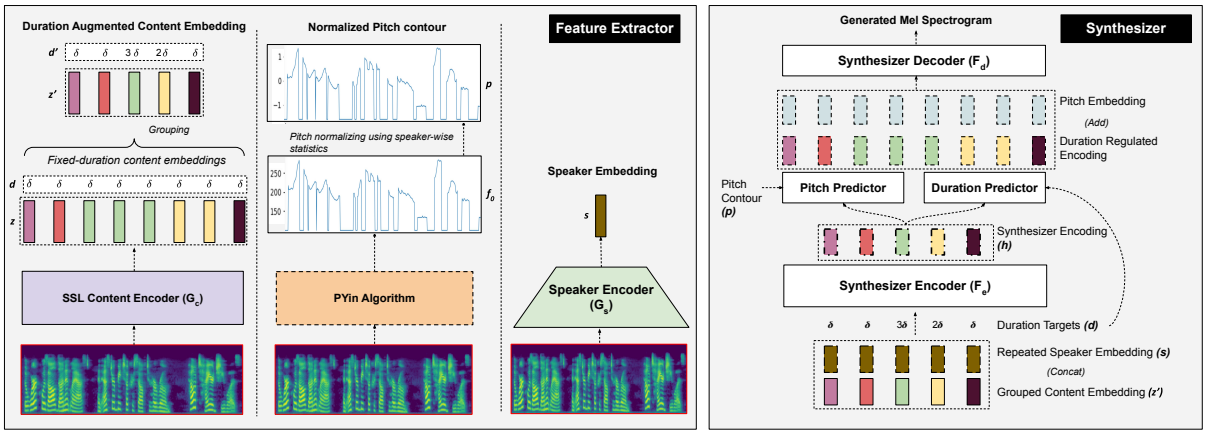


Figure 2.5. (a) The feature extractor derives the duration augmented content information from an SSL model, pitch information using PYin algorithm and speaker embedding from a speaker verification model. (b) The synthesizer reconstructs the mel-spectrogram from the derived features.

2.3.3 Synthesizer Training: Iterative Refinement using Self Transforms

While the mel-spectrogram can be accurately reconstructed from a synthesizer trained using the objective given by Equation 2.4, during inference, we cannot effectively modify the voice of a given utterance. This is because the content representation z' is not strictly disentangled from the speaker information. To address this challenge, past works [28, 29], have proposed an information perturbation based training strategy as follows: Instead of feeding the content embedding of the original audio as the input, the audio is perturbed to synthetically modify the speaker characteristics using formant-shifting, pitch-randomization and randomized frequency shaping transforms to obtain $x_p = g_{heuristic}(x)$. Next, the content embedding is derived from the perturbed audio $z' = G_c(x_p)$, while the speaker embedding is still derived from the original audio $s = G_s(x)$. The network is then tasked to reconstruct the original audio from z' and s. While heuristically perturbed content representations play a crucial role in enhancing the synthesizer model's attention towards the speaker embedding, they are limited in terms of the range of transformations they can introduce. Heuristic transformations represent only a subset of the potential natural variations that can occur during voice conversion.

To expand on the heuristic set of transforms, we propose to utilize the synthesizer model itself to generate a voice-converted variation of a given utterance x. That is, given a synthesizer model G^i_{synth} trained until training iteration i, we obtain a self transformed audio for iteration $i+1$ as:

$$x_p = g_{self}(x) = G^i_{synth}((G_c(x), s') \tag{2.5}$$

where $G_c(x)$ is the content embedding of the original audio x and s' is the speaker embedding obtained from an utterance x' of a different randomly selected speaker, that is, $s' = G_s(x')$. The content embedding input for the training step $i+1$ is then derived as $z' = G_c(x_p)$.

Self transformations not only provide a more diverse set of transformations but also present an increasingly challenging reconstruction task for the synthesizer, as its voice conversion capabilities improve with each training iteration. Figure 2.4 demonstrates the proposed self transformation training strategy. In our experiments, we begin self transformations after $100k$ mini-batch iterations of training with heuristically modified audio. Thereafter, we continue to use self transformations to obtain x_p.

2.3.4 Experiments on Voice Conversion

Dataset and Training

The Conformer-SSL model used as the content encoder is pretrained on $56k$ hours of unlabelled English speech from the LibriLight [77] corpus sampled at 16 KHz. We finetune the Conformer-SSL model (using self-supervision with contrastive and MLM loss) on the *train-clean-360* subset of LibriTTS [183] dataset with audio sampled at $22050Hz$ to make the model compatible with the mel-spectrogram representation of the synthesizer. For both the content encoder and synthesizer, we use 80 bands for mel spectrogram with the FFT, window, and hop size set to 1024, 1024, and 256 respectively. We finetune the Conformer-SSL on this revised spectrogram representation for 50 epochs with a batch size of 32 using the AdamW optimizer

with a fixed learning rate of $5e-5$ and $\beta_1 = 0.9, \beta_2 = 0.99$. Finetuning takes around 50 hours on a single NVIDIA A600 GPU.

For our primary experiments, the mel-spectrogram synthesizer and the HifiGAN vocoder are also trained on the *train-clean-360* subset of the LibriTTS dataset which contains 360 hours of speech from 904 speakers. We train three variants of the mel-spectrogram synthesizer:

1. **Synth (NoTransform)** is trained to simply reconstruct the mel-spectrogram from the embeddings of the given utterance without any information perturbation procedure.

2. **Synth (Heuristic)** is trained to reconstruct the mel-spectrogram from the content embedding of the heuristically perturbed utterance and the speaker embedding of the original utterance. We employ two transforms g_1, g_2 proposed in [28]. g_1 perturbs formant, pitch, and frequency response and g_2 perturbs formant and frequency response while preserving pitch.

3. **Synth (SelfTransform)** is first trained in the same way as Synth-Heuristic for the first $100k$ mini batch iterations. Thereafter, we use the g_{self} transformation procedure given by Equation 2.5.

All three variants of the synthesizer are optimized using AdamW optimizer [106] with a fixed learning rate of $1e-4$ and $\beta_1 = 0.8, \beta_2 = 0.99$ for 500 epochs with a batch size of 32. The threshold τ for duration extraction is set as 0.925. The loss coefficients for the duration and pitch loss are set as $\lambda_1 = \lambda_2 = 0.1$. Training time for Synth (SelfTransform) model is around 5 days on 4 NVIDIA A600 GPUs. The HifiGAN vocoder is trained on the mel-spectrogram and waveform pairs of the real audio utterances and the same vocoder is used across all three synthesizers.

Evaluation Metrics

Quantitatively, we evaluate the synthesized audio on the following aspects:

1. **Intelligibility (CER):** We compute the Character Error Rate (CER) between the ASR transcriptions of the original source and the generated audio. We use pre-trained Quartznet [90]

ASR models for the respective language of the given utterance.

2. **Speaker Similarity (SV-EER):** To evaluate speaker similarity to our target speaker, we compute the speaker embeddings of synthesized and real utterances using a separate pre-trained speaker verification model [86]. Then we pair the synthesized and real utterances to create an equal number of positive and negative pairs for each target speaker to compute the Equal Error Rate (SV-EER).

3. **Naturalness (MOS):** We perform a human study on Amazon Mechanical Turk, where human judges rate the naturalness of each utterance on a 1 to 5 scale with 0.5 point increments. Each utterance is rated by 4 independent listeners and each listener can rate multiple utterances. For 200 synthesized utterances from each technique, this procedure results in a total of 800 evaluations of each technique.

4. **Prosodic Similarity (GPE):** To evaluate prosodic similarity for the reconstruction task (Section 2.3.4), we compute the error between the fundamental frequency contours of the original and synthesized audio. Specifically, we use the Gross Pitch Error (GPE) [34] to evaluate prosodic similarity.

Reconstruction Results

First, we evaluate how effectively our setup can reconstruct audio from the extracted representations for unseen utterances and speakers. Our synthesizers can operate in two modes during inference — *1) Guided:* In this scenario, we use ground truth pitch and duration information derived from the source utterance. *2) Predictive:* In this case, we use the predicted pitch and duration for synthesis.

We conduct the reconstruction test on two unseen datasets — 1) We choose 200 utterances from the VCTK [178] dataset (English) with 20 random utterances from each of the 10 speakers (5 random male and 5 random female speakers); 2) To evaluate performance on unseen languages, we choose 200 utterances from the CSS10 [126] dataset with 20 random utterances from each

of the 10 unseen languages. The CSS10 dataset has a single speaker per language. For both of these evaluations, we use the synthesizer models trained on the same dataset, i.e. *train-clean-360* subset of LibriTTS (English). The synthesized speech is evaluated on the intelligibility, speaker similarity and prosodic similarity metrics. As indicated by the results in Table 2.4, all three synthesizers can effectively reconstruct the speech signal from the derived representation. Since the model is trained in a text-free manner, we also see a promising generalization to unseen languages. For unseen languages, our synthesizers produce more intelligible speech in the guided mode, where the duration information of the source utterance is kept intact. In the reconstruction mode, since the speaker and content embeddings are derived from the same utterance, both Synth (NoTransform) and Synth (Heuristic) models achieve competitive speaker similarity to the target speaker. However, for controllable synthesis tasks such as voice conversion, we demonstrate that Synth (SelfTransforms) outperforms these models.

Table 2.4. Reconstruction evaluation: The resynthesized speech from different synthesizers is evaluated for intelligibility (CER), speaker similarity (SV-EER) and prosodic similarity (GPE). Lower values are desirable for all three metrics.

		Guided			*Predictive*		
Dataset	Technique	SV-EER	CER	GPE	SV-EER	CER	GPE
	Real Data	3.1%	-	-	3.1%	-	-
VCTK	Synth (NoTransform)	4.6%	3.5%	8.0%	4.7%	4.9%	22.0%
(English)	Synth (Heuristic)	4.3%	2.9%	8.8%	4.5%	4.1%	21.1%
Seen Language	Synth (SelfTransform)	4.2%	2.2%	8.9%	4.1%	3.9%	21.0%
	Real Data	2.3%	-	-	2.3%	-	-
CSS10	Synth (NoTransform)	5.5%	25.5%	11.7%	4.9%	29.8%	15.9%
(Multilingual)	Synth (Heuristic)	5.3%	26.1%	11.6%	5.5%	30.2%	16.1%
Unseen Language	Synth (SelfTransform)	4.1%	25.2%	10.8%	4.8%	29.2%	16.8%

Voice Conversion Results

To convert the voice of a given source utterance to a target speaker, we derive the content embedding from the source utterance and estimate the speaker embedding from the target speaker's audio and feed both as input to the synthesizer. We consider two voice conversion scenarios — for a seen speaker to another seen speaker from the training data (Many-to-Many)

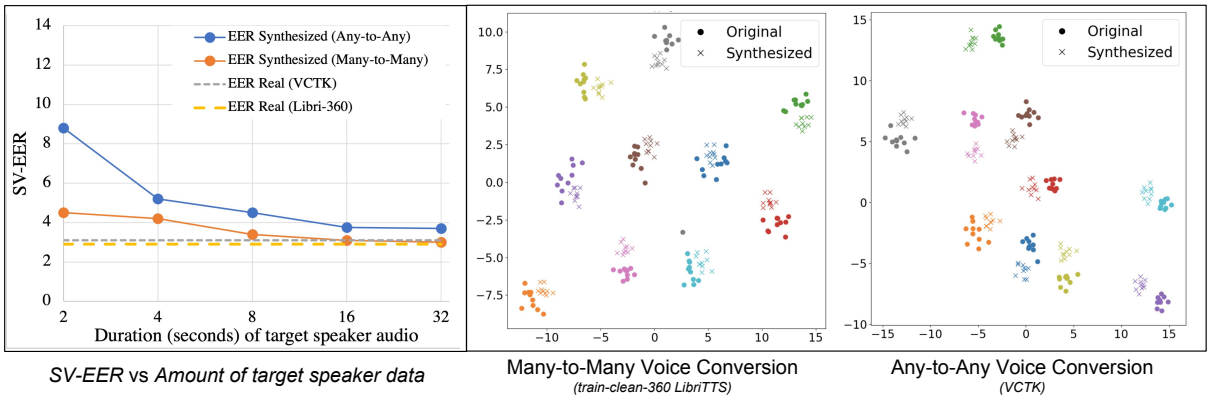
and from an unseen speaker to another unseen speaker outside of training data (Any-to-Any). For seen speakers, we use the holdout utterances of the *train-clean-360* subset of LibriTTS dataset, and for unseen speakers, we use the VCTK dataset. For each scenario, we randomly select 20 target speakers (10 male and 10 female). Next, we select 10 source utterances, each one from 10 alternate speakers. This results in a total of 200 voice conversion trials in each scenario.

Table 2.5. Comparison of different voice-conversion techniques. Lower values for SV-EER and CER are desirable for higher speaker similarity and intelligibility respectively. Higher MOS (reported with 95% confidence interval) indicates more natural-sounding speech.

	Many-to-Many			*Any-to-Any*		
Technique	SV-EER	CER	MOS	SV-EER	CER	MOS
Real Data	2.9%	-	4.03 ± 0.09	3.1%	-	4.08 ± 0.09
AutoVC [134]	23.5%	21.2%	2.75 ± 0.11	38.3%	34.2%	2.46 ± 0.11
AdaIN-VC [33]	18.2%	29.2%	2.64 ± 0.11	27.5%	30.3%	2.82 ± 0.12
MediumVC [60]	10.2%	31.5%	3.01 ± 0.12	23.2%	36.2%	2.95 ± 0.11
FragmentVC [101]	15.9%	27.2%	3.10 ± 0.11	24.8%	38.5%	3.11 ± 0.12
S3PRL-VC [68]	13.7%	9.8%	3.20 ± 0.11	22.8%	9.8%	3.14 ± 0.12
YourTTS [23]	9.5%	6.1%	3.52 ± 0.10	12.3%	7.9%	3.58 ± 0.10
ACE-VC [71]	5.3%	3.7%	3.58 ± 0.10	9.2%	8.2%	3.68 ± 0.09
Synth (NoTransform)	19.1%	2.6%	3.55 ± 0.12	25.2%	3.8%	3.51 ± 0.11
Synth (Heuristic)	4.4%	2.3%	3.69 ± 0.12	10.5%	3.1%	3.65 ± 0.12
Synth (SelfTransform)	**3.0%**	**2.2%**	$\mathbf{3.72 \pm 0.11}$	**4.3%**	**3.1%**	$\mathbf{3.75 \pm 0.11}$

For our primary evaluation, we use 10 seconds of speech from each target speaker to derive the speaker embedding. We split the 10 second target-speaker utterance into 2 second segments and estimate the speaker embedding as the mean speaker embedding across the segments. We also evaluate the speaker-similarity performance for different amounts of target speaker data and present the results in Figure 2.6.

The synthesized speech is evaluated on three aspects: speaker similarity, intelligibility and naturalness. We compare our synthesis model against several prior voice conversion methods listed in Table 2.5. While NANSY [29] is not open-sourced, our Synth (Heuristic) baseline model closely follows the training strategy proposed in NANSY, incorporating more recent neural architectures for the synthesizer and feature extractors. As shown by the results, the Synth



SV-EER vs *Amount of target speaker data*

Many-to-Many Voice Conversion
(train-clean-360 LibriTTS)

Any-to-Any Voice Conversion
(VCTK)

Figure 2.6. Left: SV-EER of voice-converted speech generated by Synth (SelfTransform) using different amounts of target speaker data. **Right:** TSNE visualization of speaker embeddings of generated (using Synth (SelfTransform)) and ground-truth audio. Each color represents a different speaker.

(SelfTransform) model outperforms the Synth (NoTransform) and Synth (Heuristic) models on the speaker similarity metrics. The improvement is even more notable for *Any-to-Any* voice conversion task. On all three metrics, our proposed technique outperforms previously proposed voice conversion models. In Figure 2.6, we show TSNE plots of the speaker embeddings of generated and real audio.

Cross-lingual Voice Conversion: In this setup, we consider three scenarios — 1) **S2U:** Source utterance from a seen language speaker (English VCTK) and target speaker from an unseen language (CSS10). 2) **U2S:** Source utterance from an unseen language (CSS10) and target speaker from the source language (English VCTK). 3) **U2U:** Source utterance from an unseen language (CSS10) and target speaker from another unseen language (CSS10).

We present the results in Table 2.6. While the Synth (SelfTransform) model generates speech with high speaker-similarity in all three scenarios, the generated speech is more intelligible when the source utterance is in English. This is not surprising since the synthesizer is trained on only English speech (LibriTTS).

Table 2.6. Results on cross-lingual voice conversion task in three scenarios considering different languages for source utterance and target speaker. Lower SV-EER is desirable for higher speaker similarity and lower CER is desirable for more intelligible speech.

	S2U		U2S		U2U	
Technique	SV-EER	CER	SV-EER	CER	SV-EER	CER
Real Data	2.3%	-	3.1%	-	3.1%	-
Synth (NoTransform)	31.2%	3.8%	28.2%	29.7%	39.1%	30.7%
Synth (Heuristic)	15.3%	3.1%	9.0%	28.5%	22.1%	29.5%
Synth (SelfTransform)	8.5%	3.0%	5.4%	27.5%	15.1%	29.1%

2.4 Conclusion

In this chapter, I described two speech synthesis frameworks that tackle the problem of expressive speech synthesis for new speakers. The first framework is a voice-cloning method that performs text-to-speech synthesis with explicit control over speaker and style aspects. By utilizing both latent and heuristically derived style information, the model is able to learn a wide-range style control for unseen speakers while being trained on a mostly style-neutral dataset. The second framework is a voice conversion model that proposes a training strategy to perform controllable speech synthesis from imperfectly disentangled speech representations. The synthesis model of the voice conversion framework allows speaker-adaptive duration and pitch control for more natural-sounding speech achieving state-of-the-art results on various voice conversion metrics. Both of the above frameworks enable high-quality speech synthesis that can accompany AI-generated visuals for various generative media applications.

2.5 Acknowledgements

Chapter 2 contains material found in the following two papers. (1) *Expressive Neural Voice Cloning*. 2021. Neekhara, Paarth; Hussain, Shehzeen; Dubnov, Shlomo; Koushanfar, Farinaz; McAuley, Julian. Asian Conference on Machine Learning 2021. (2) *Controllable Speech Synthesis with Iterative Refinement using Self Transformations*. 2023. Neekhara, Paarth;

Hussain, Shehzeen; Ranjan, Rishabh; Dubnov, Shlomo; Koushanfar, Farinaz; McAuley, Julian. Currently under review for publication. The dissertation author was the primary investigator and author of these papers.

Chapter 3

Synthetic Media Detectors and Their Vulnerability to Adversarial Attacks

Deep Neural Networks (DNNs) have brought about a significant advancement in the realm of digital media generation. DNNs have not only improved the quality of artificially generated and forged media, but they have also rendered the process of creating fake content much simpler. Deepfakes are a new genre of synthetic videos, in which a subject's face is modified into a target face in order to simulate the target subject in a certain context and create convincingly realistic footage of events that never occurred. Video manipulation methods like Face2Face [162], Neural Textures [161] and FaceSwap [89] operate end-to-end on a source video and target face and require minimal human expertise to generate fake videos in real-time. Combined with the voice cloning and voice conversion techniques described in Chapter 2, these methods can create high-quality fake videos that are hard to distinguish from real videos.

The intent of generating such videos can be harmless and has led to advances in research on synthetic video generation for movies, storytelling, and modern-day streaming services. However, they can also be used maliciously to spread misinformation, harass individuals or defame famous personalities [156]. These videos are now an emerging threat, especially within the realms of politics and misinformation. Deepfakes have been used to create fake news aggravating political and religious tensions, with the aim to influence results in election campaigns [63, 76, 120]. Such extensive spread of fake videos through social media platforms has

raised significant concerns worldwide, particularly hampering the credibility of digital media. Recent research has found evidence that widespread misinformation not only misleads individuals and reduces public trust on digital media but also leads to increased cynicism within democratic societies [166].

To address the threats imposed by Deepfakes, the machine learning community has proposed several countermeasures to identify forgeries in digital media [169]. In this chapter, we first discuss the recently proposed state-of-the-art methods to detect Deepfake videos. The recently proposed state-of-the-art methods use a visual DNN-based classification system that is trained in a supervised manner on a curated dataset of real and fake videos. Deepfake detection is typically modeled as a per-frame classification problem. Additionally, the best performing models employ a face-tracking method following which the cropped face from a frame is passed on to a CNN-based classifier for classification as real or fake [2, 32, 143, 65]. Some of the recent Deepfake detection methods also use models that operate on a sequence of frames as opposed to a single frame to exploit temporal dependencies in videos [40].

While the above DNN-based detection methods achieve promising results in accurately detecting manipulated videos, my work uncovers major vulnerabilities in such systems. Later in the chapter, I describe my work on *AdversarialDeepfakes* that examines the vulnerabilities of Deepfake detection systems adversarial examples. An adversarial example is an intentionally perturbed input that can fool a victim classification model [157]. We quantitatively assess the vulnerability of Deepfake detectors to adversarial examples in different threat scenarios. Assuming a complete access (white-box) threat scenario, we find that it is trivial to bypass a Deepfake detector with an imperceptible adversarial modification to a given video. However, in a practical threat scenario the attacker may not have knowledge of the victim detection model and parameters. To this end, we assume a more challenging threat scenario in which the attacker can only query a victim model to get the detection scores for a video frame. Even in this attack scenario, we find that it is possible to bypass the detector with a slightly higher amount of adversarial perturbation. Additionally, to ensure the adversarial videos remain effective even

after video compression, we incorporate expectation over input transforms [5] while training the adversarial perturbation to craft robust adversarial videos. While the above attacks can effectively bypass Deepfake detectors, they can be easily thwarted by the service provider. Detection models and parameters can be kept private to prevent the white-box attack and query access can be limited to prevent the black-box attack. Adversarial examples pose a practical threat to Deepfake detection if they are transferable across different detection methods. That is, if adversarial videos designed to fool some open source Deepfake detection method can also reliably fool other unseen CNN-based detection methods, this would pose a real security threat to deploying CNN-based detectors in production. We experimentally demonstrate that it is possible to design highly transferable adversarial examples by ensuring robustness to input-transformation functions while training the perturbation. Finally, we design more accessible adversarial attacks by creating transferable universal adversarial perturbations that can be universally added across all frames of all videos to reliably fool a number of Deepfake detection methods.

3.1 Deepfake Detection Datasets

Deepfake detection methods rely on the availability of high-quality deepfake detection datasets, which are crucial for training and evaluating deepfake detection models. Several Deepfake detection datasets have been developed in recent years, each with its own unique characteristics and properties. One of the most widely used Deepfake detection datasets is the FaceForensics++ dataset. This dataset contains Deepfake and real videos captured using a variety of cameras and settings. The dataset includes four types of Deepfakes:

- **FaceSwap (FS):** FaceSwap [89] is a classical computer graphics-based approach for face replacement in videos. In this method, sparse facial landmarks are detected to extract the face region in an image. These landmarks are then used to fit a 3D template model which is back-projected onto the target image by minimizing the distance between the projected shape and localized landmarks. Finally, the rendered model is blended with the image and

color correction is applied.

- **Face2Face (F2F):** Face2Face [162] is a facial reenactment system that transfers the expressions of a person in a source video to another person in a target video, while maintaining the identity of the target person. In this method, faces are compressed into a low-dimensional expression space, where expressions can be easily transferred from the source to the target.

- **DeepFakes (DF):** While the term 'Deepfake' has commonly been used in mainstream media as a blanket term for deep-learning based face replacement, it is also the name of a specific manipulation [42] method that was spread via online forums. In the learning phase, two auto-encoders with a shared encoder are trained to reconstruct the images of source and target face. To create a fake image, the encoded source image is passed as input to the target image decoder.

- **NeuralTextures (NT):** NeuralTextures [161] is a Generative Adversarial Network (GAN) based facial reenactment technique. In this method, a generative model is trained to learn the neural texture of a target person using original video data. The GAN objective is a combination of an adversarial and photometric reconstruction loss.

FaceForensics++ has been widely used in research, with several deepfake detection models trained on this dataset. Aside from the FaceForensics++ dataset, another prominent collection of Deepfake videos was released by Facebook, Inc in 2019. To the best of our knowledge, this recently developed DeepFake Detection Challenge (DFDC) dataset [46] is the largest collection of real and Deepfake videos, consisting of over one million training clips of face swaps produced with a variety of methods. For synthesizing the fake videos in the DFDC dataset, 8 different video manipulation techniques were used, many of which are CNN-based techniques. These methods include the traditional Deepfake auto-encoder architecture, a non-learned morphable mask face swap algorithm, and several Generative Adversarial Networks

(GAN) techniques like Neural Talking Heads [182], FSGAN [121] and StyleGAN [81]. In conjunction with the dataset, a corresponding competition[1] was launched in which competitors were encouraged to submit models trained for Deepfake detection on the training set. These models were then ranked on a hidden, held-out test set, and the winning competitors released their architectures and training strategies publicly.

3.2 Deepfake Detectors

Traditionally, multimedia forensics investigated the authenticity of images [172, 18, 52] using hand-engineered features and/or a-priori knowledge of the statistical and physical properties of natural photographs. However, video synthesis methods can be trained to bypass hand-engineered detectors by modifying their training objective. We direct readers to [11, 19] for an overview of counter-forensic attacks to bypass traditional (non-deep learning based) methods of detecting forgeries in multimedia content.

More recent works have employed CNN-based approaches that decompose videos into frames to automatically extract salient and discriminative visual features pertinent to Deepfakes. Some efforts have focused on segmenting the entire input image to detect facial tampering resulting from face swapping [189], face morphing [136] and splicing attacks [9, 10]. Other works [98, 99, 2, 61, 139, 140] have focused on detecting face manipulation artifacts resulting from Deepfake generation methods. The authors of [99] reported that eye blinking is not well reproduced in fake videos, and therefore proposed a temporal approach using a CNN + Recurrent Neural Network (RNN) based model to detect a lack of eye blinking when exposing Deepfakes. Similarly, [180] used the inconsistency in head pose to detect fake videos. However, this form of detection can be circumvented by purposely incorporating images with closed eyes and a variety of head poses in training [170, 50].

[1] https://www.kaggle.com/c/deepfake-detection-challenge

3.2.1 Per-frame Deepfake Detectors

The Deepfake detectors proposed in [139, 2, 46] model Deepfake detection as a per-frame binary classification problem. The authors of [139] demonstrated that XceptionNet can outperform several alternative classifiers in detecting forgeries in both uncompressed and compressed videos, and identifying forged regions in them. Since the task is to specifically detect facial manipulation, these models incorporate domain knowledge by using a face tracking method [162] to track the face in the video. The face is then cropped from the original frame and fed as input to a classification model to be labelled as real or fake. Experimentally, the authors of [139] demonstrate that incorporation of domain knowledge helps improve classification accuracy as opposed to using the entire image as input to the classifier. The best performing classifiers amongst others studied by [139] were both CNN based models: XceptionNet [32] and MesoNet [2]. Figure 3.1 demonstrates the detection pipeline of these per-frame Deepfake classifiers.

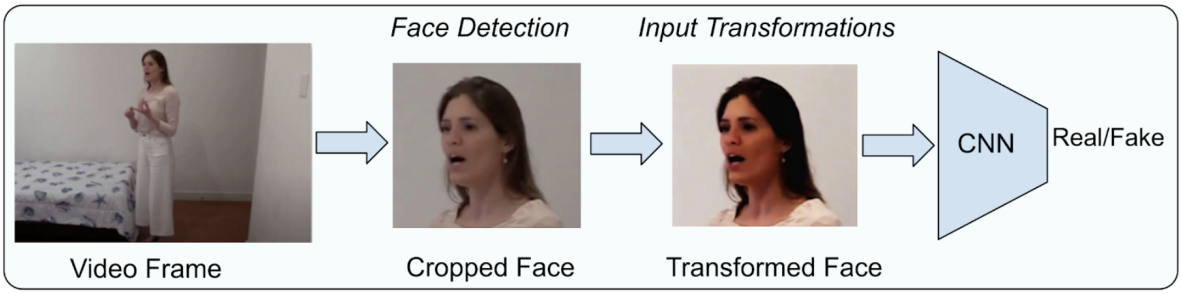


Figure 3.1. Per-frame Deepfake Classification Models typically follow a two-step pipeline: Face detection followed by binary classification.

3.2.2 Sequence-based Deepfake Classifiers

Some detectors have also focused on exploiting temporal dependencies for detecting Deepfake videos. Such detectors work on sequence of frames as opposed to a single frame using a CNN + RNN model or a 3D CNN model. One such model based on a 3D EfficientNet [159] architecture, was used by the third place winner [40] of DFDC challenge [46] in addition to

a per-frame classification model. While intuitively, exploiting temporal dependencies using sequence models should improve a detector's ability to detect manipulated videos, the insights from the results of the DFDC challenge [46] show that the best performing models operate on a frame level. In fact, the winning team [143] of the DFDC challenge explicitly noted that other ideas besides frame-by-frame detection did not improve their performance on the public leaderboard. The first two winning submissions were both CNN based per-frame classification models similar to the ones described above.

3.2.3 Understanding Deepfake detectors

To gain insight into the decision-making logic of Deepfake detectors, we obtain the gradient of the score of the predicted class with respect to the input image and plot the magnitude of these gradients as a heat-map. Back-propagating gradients naively does not result in very interpretable visualizations. This is because it is more important to consider pixels which activate a neuron and do not suppress it (suppression is indicated by negative gradients). Therefore, we use guided back-propagation which defines custom gradient estimates for activation functions like ReLU and suppresses negative gradients during the backward pass. We then standardize the gradient obtained with respect to the input and overlay the heat-map on the frame to visualize the areas of an image that trigger the network's output. Figure 3.2 shows some examples of the saliency maps obtained while analyzing two different detectors on Deepfake videos.

Our initial observations on these saliency maps suggest that different CNN based detection methods attend to similar aspects of the input frame for predicting the label. These aspects include the edges of the face, the eyes, lips, teeth etc. These similarities across different detection methods indicate that adversarially modifying such aspects of the image could potentially fool multiple detection methods. We validate this hypothesis in our work by studying the transferability of adversarial examples (Section 3.5.2) across different detection methods and proposing techniques (Section 3.3.3) that improve the transferability.

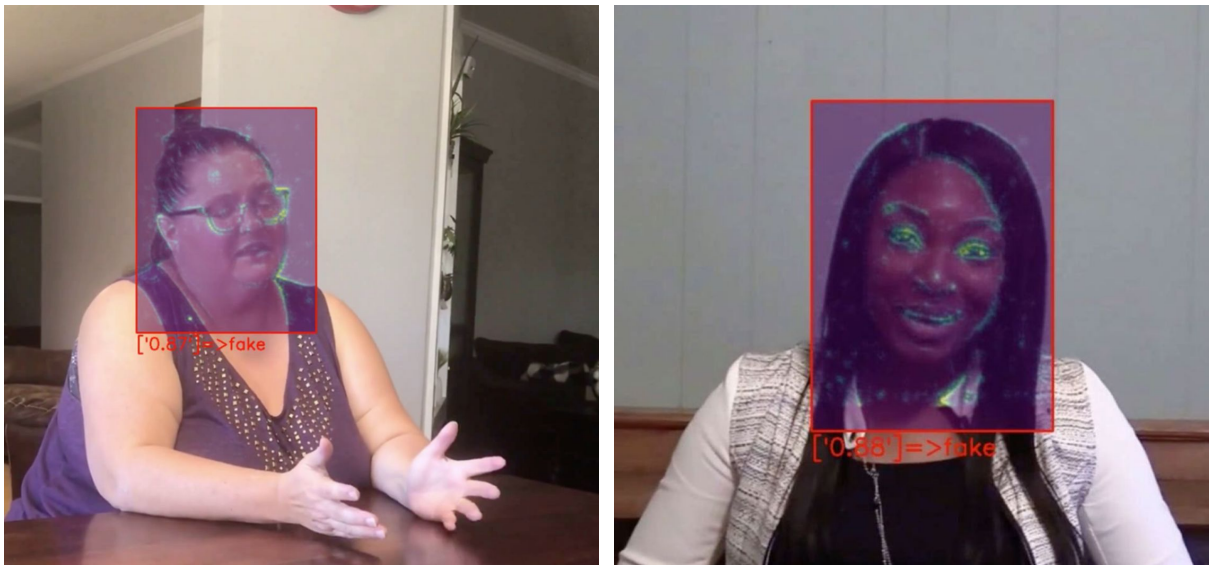


Figure 3.2. Gradient saliency maps obtained on Deepfake videos using guided backpropogation on a CNN-based detector [143]. The highlighted areas indicate the image regions that strongly influence the detector's predictions.

3.3 Adversarial attacks on Deepfake detectors

In this section, I discuss the threat models for Deepfake detectors in various attack settings assuming different attacker capabilities. First, we mathematically define the threat model and attack goal (Section 3.3.1). Next, we propose a white-box attack to achieve the attack goal in a scenario when the attacker has complete access to the victim model architecture and parameters (Section 3.3.2). In our experiments, we find that while the simple white-box attack works well on uncompressed videos, the attack success rate drops significantly on compressed videos. Another challenge in the simple white-box attack is the limited transferability of the attack to unseen models. We tackle these two challenges using our robust and transferable attack which poses a real world threat — the adversarial videos are more robust to compression and can also fool unseen detectors to a significant extent thereby posing a real-world threat (Section 3.3.3). Next we propose query based black-box attacks which do not require access to any surrogate model but only require query access to the model scores (Section 3.3.4, 3.3.5). Finally, we propose a highly accessible attack using universal adversarial perturbations — we find that it is possible to

craft a single input-agnostic perturbation that can be added across all frames of any given video to cause classification to the target label by many seen and unseen detectors. Once crafted, this perturbation can be easily shared amongst adversaries thereby posing a very practical challenge to Deepfake detection (Section 3.3.6).

3.3.1 Threat Model

Given a video (*Real* or *Fake*), our task is to adversarially modify the video such that the label predicted by a victim Deepfake detection method is incorrect. That is, we want to modify the videos such that the *Fake* videos are classified as *Real* and vice-versa. Misclassifying a *Fake* video as *Real* can be used by the adversary to propagate false information. Misclassifying a *Real* video as *Fake* can be used by the adversary to cover up an event that did actually happen.

Distortion Metric

To ensure imperceptibility of the adversarial modification, the L_p norm is a widely used distance metric for measuring the distortion between the adversarial and original inputs. The authors of [57] recommend constraining the maximum distortion of any individual pixel by a given threshold ε, i.e., constraining the perturbation using an L_∞ metric. Additionally, *Fast Gradient Sign Method* (FGSM) [57] based attacks, which are optimized for the L_∞ metric, are more time-efficient than attacks which optimize for L_2 or L_0 metrics. Since each video can be composed of thousands of individual frames, time-efficiency becomes an important consideration to ensure the proposed attack can be reliably used in practice. Therefore, in this work, we use the L_∞ distortion metric for constraining our adversarial perturbation and optimize for it using gradient sign based methods.

Notation

We follow the notation previously used in [21, 125]; we define F to be the full neural network (classifier) including the softmax function, $Z(x) = z$ to be the output of all layers except

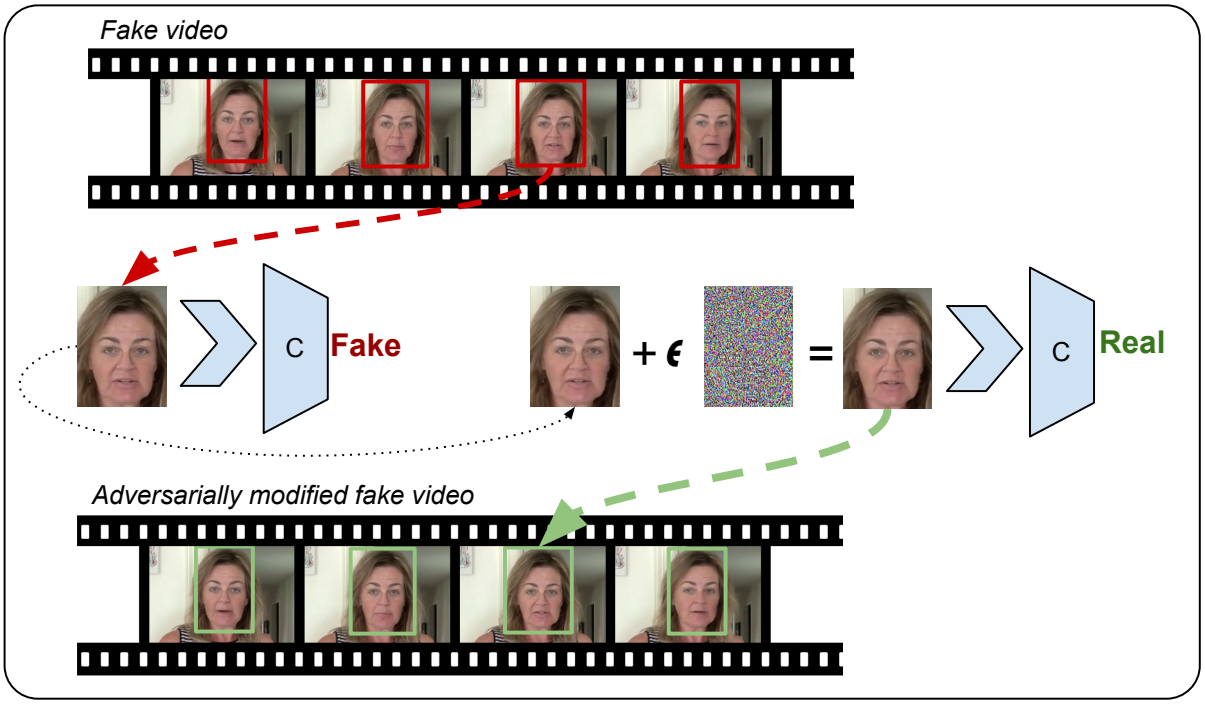


Figure 3.3. An overview of our attack pipeline to generate Adversarial Deepfakes. We generate an adversarial example for each frame in the given fake video and combine them together to create an adversarially modified fake video.

the softmax (that is z are the logits). That is:

$$F(x) = softmax(Z(x)) = y$$

The classifier assigns the label $C(x) = \arg\max_i(F(x)_i)$ to input frame x.

Problem Formulation

Mathematically, for each video frame x, we aim to find an adversarial frame x_{adv} such that:

$$C(x_{adv}) = y \text{ and } ||x_{adv} - x_0||_\infty < \varepsilon$$

where y is the target label. In our case the target label is *Real* for *Fake* videos and *Fake* for *Real* videos. In the upcoming sections, we study this attack goal in various attacker knowledge settings and constraints.

Attack Pipeline

An overview of the process of generating adversarial fake videos is depicted in Figure 3.3. For any given frame, we craft an adversarial example for the cropped face, such that after going through some image transformations (normalization and resizing), it gets classified as *Real* by the classifier. The adversarial face is then placed in the bounding box of face-crop in the original frame, and the process is repeated for all frames of the video to create an adversarially modified fake video. In the following sections, we consider our attack pipeline under various settings and goals. Note that, the proposed attacks can also be applied on detectors that operate on entire frames as opposed to face-crops. We choose face-crop based victim models because they have been shown to outperform detectors that operate on entire frames for detecting facial-forgeries.

3.3.2 Simple White-box attack

In this setting, we assume that the attacker has complete access to the detection model, including the face extraction pipeline and the architecture and parameters of the classification model. To construct adversarial examples using the attack pipeline described above, we use the iterative gradient sign method [91] to optimize the following objective:

$$\begin{aligned} &\text{Minimize } loss(x') \text{ where} \\ &loss(x') = max(Z(x')_o - Z(x')_y, 0) \end{aligned} \tag{3.1}$$

Here, $Z(x)_y$ is the final score for target label y and $Z(x)_o$ is the score of the original label o before the softmax operation in the classifier C. The loss function we use is recommended by [21] because it is empirically found to generate less distorted adversarial samples and is robust against defensive distillation. We use the iterative gradient sign method to optimize the above loss function while constraining the magnitude of the perturbation as follows:

$$x_i = x_{i-1} - \text{clip}_\varepsilon(\alpha \cdot \text{sign}(\nabla loss(x_{i-1}))) \tag{3.2}$$

We continue gradient descent iterations until success or until a given number of maximum iterations, whichever occurs earlier. We solve the optimization problem for each frame of the given video and combine all the adversarial frames together to generate the adversarial video. In our experiments, we demonstrate that we are able to successfully fool all the detection methods studied in our work in the white-box attack setting using the above attack. However, the transferability of adversarial examples generated using this attack across different methods is limited. In the next section we propose techniques to overcome this challenge.

3.3.3 Robust and Transferable attack

Videos uploaded to social networks and other media sharing websites are usually compressed. Standard operations like compression and resizing are known to remove adversarial perturbations from an image [51, 39, 64]. To ensure that the adversarial videos remain effective even after compression, it is important to ensure robustness to input-transformation functions while training the perturbation.

Also, past works [48, 49, 103, 124, 177, 190] have studied that adversarial inputs can transfer across different models. That is, an adversarial input that was designed to fool a particular victim model can possibly fool other models that were trained for the same task. This is because different models learn similar decision boundaries and therefore have similar vulnerabilities. However, for Deepfake detectors, the goal of making transferable adversarial videos is more challenging due to multiple steps involved in the Deepfake detection pipeline and the differences in these steps across various methods.

- Different face detection methods result in different face-crops.
- Different data-augmentation procedures during training result in different levels of robustness to adversarial examples.
- Different input pre-processing pipelines, such as image resizing, cropping and channel normalization parameters vary across different detection methods.

Therefore ensuring robustness to input transformation functions not only helps create adversarial videos that are robust to compression, but can also potentially result in adversarial videos that are transferable across different detection methods. We use the expectation over transforms [5] attack to craft robust and transferable adversarial examples. Given a distribution of input transformations T, input image x, and target class y, our objective is as follows:

$$x_{adv} = argmax_x \mathbb{E}_{t\sim T}[F(t(x))_y] \text{ s.t. } ||x - x_0||_\infty < \varepsilon$$

That is, we want to maximize the expected probability of target class y over the distribution of input transforms T. To solve the above problem, we update the loss function given in Equation 3.1 to be an expectation over input transforms T as follows:

$$loss(x) = \mathbb{E}_{t\sim T}[max(Z(t(x))_o - Z(t(x))_y, 0)]$$

Following the law of large numbers, we estimate the above loss functions for n samples as:

$$loss(x) = \frac{1}{n}\sum_{t_i\sim T}[max(Z(t_i(x))_o - Z(t_i(x))_y, 0)] \tag{3.3}$$

Since the above loss function is a sum of differentiable functions, it is tractable to compute the gradient of the loss w.r.t. to the input x. We minimize this loss using the iterative gradient sign method given by Equation 3.2. We iterate until a given number of maximum iterations or until the attack is successful under the sampled set of transformation functions, whichever happens first.

Next we describe the class of input transformation functions we consider for the distribution T:

- **Gaussian Blur:** Convolution of the original image with a Gaussian kernel k. This transform is given by $t(x) = k * x$ where $*$ is the convolution operator.

- **Gaussian Noise Addition:** Addition of Gaussian noise sampled from $\Theta \sim \mathcal{N}(0, \sigma)$ to the input image. This transform is given by $t(x) = x + \Theta$
- **Translation:** We pad the image on all four sides by zeros and shift the pixels horizontally and vertically by a given amount. Let t_x be the transform in the x axis and t_y be the transform in the y axis, then $t(x) = x'_{H,W,C}$ s.t. $x'[i, j, c] = x[i + t_x, j + t_y, c]$
- **Downsizing and Upsizing:** The image is first downsized by a factor r and then up-sampled by the same factor using bilinear re-sampling.

The details of the hyper-parameter search distribution used for these transforms can be found in the Section 3.5.1.

3.3.4 Query based Black-box Attack

In the black-box setting, we consider the more challenging threat model in which the adversary does not have access to the classification network architecture and parameters. We assume that the attacker has knowledge of the detection pipeline structure and the face tracking model. However, the attacker can solely query the classification model as a black-box function to obtain the probabilities of the frame being *Real* or *Fake*. Hence there is a need to estimate the gradient of the loss function by querying the model and observing the change in output for different inputs, since we cannot backpropagate through the network.

We base our algorithm for efficiently estimating the gradient from queries on the Natural Evolutionary Strategies (NES) approach of [175, 72]. Since we do not have access to the pre-softmax outputs Z, we aim to maximize the class probability $F(x)_y$ of the target class y. Rather than maximizing the objective function directly, NES maximizes the expected value of the function under a search distribution $\pi(\theta|x)$. That is, our objective is:

$$\textit{Maximize: } \mathbb{E}_{\pi(\theta|x)}[F(\theta)_y]$$

This allows efficient gradient estimation in fewer queries as compared to finite-difference methods. From [175], we know the gradient of expectation can be derived as follows:

$$\nabla_x \mathbb{E}_{\pi(\theta|x)}\left[F(\theta)_y\right] = \mathbb{E}_{\pi(\theta|x)}\left[F(\theta)_y \nabla_x \log\left(\pi(\theta|x)\right)\right]$$

Similar to [72, 175], we choose a search distribution $\pi(\theta|x)$ of random Gaussian noise around the current image x. That is, $\theta = x + \sigma\delta$ where $\delta \sim \mathcal{N}(0,I)$. Estimating the gradient with a population of n samples yields the following variance reduced gradient estimate:

$$\nabla\mathbb{E}[F(\theta)] \approx \frac{1}{\sigma n}\sum_{i=1}^{n} \delta_i F(\theta + \sigma\delta_i)_y$$

We use antithetic sampling to generate δ_i similar to [142, 72]. That is, instead of generating n values $\delta \sim \mathcal{N}(0,I)$, we sample Gaussian noise for $i \in \{1,\ldots,\frac{n}{2}\}$ and set $\delta_j = -\delta_{n-j+1}$ for $j \in \{(\frac{n}{2}+1),\ldots,n\}$. This optimization has been empirically shown to improve the performance of NES. Algorthim 2 details our implementation of estimating gradients using NES. The transformation distribution T in the algorithm just contains an identity function i.e., $T = \{I(x)\}$ for the black-box attack described in this section.

After estimating the gradient, we move the input in the direction of this gradient using iterative gradient sign updates to increase the probability of the target class:

$$x_i = x_{i-1} + \text{clip}_\varepsilon(\alpha \cdot \text{sign}(\nabla F(x_{i-1})_y)) \tag{3.4}$$

3.3.5 Query based Robust Black-box Attack

To ensure robustness of adversarial videos to compression, we incorporate the Expectation over Transforms (Section 3.3.3) method in the black-box setting for constructing adversarial videos.

To craft adversarial examples that are robust under a given set of input transformations T, we maximize the expected value of the function under a search distribution $\pi(\theta|x)$ and our distribution of input transforms T. That is, our objective is to maximize:

$$\mathbb{E}_{t\sim T}[\mathbb{E}_{\pi(\theta|x)}[F(t(\theta))_y]]$$

Following the derivation in the previous section, the gradient of the above expectation can be estimated using a population of size n by iterative sampling of t_i and δ_i:

$$\nabla\mathbb{E}[F(\theta)] \approx \frac{1}{\sigma n}\sum_{i=1,t_i\sim T}^{n}\delta_i F(t_i(\theta+\sigma\delta_i))_y$$

Algorithm 2. NES Gradient Estimate

Input: Classifier $F(x)$, target class y, image x
Output: Estimate of $\nabla_x F(x)_y$
Parameters: Search variance σ, number of samples n, image dimensionality N
$g \leftarrow 0_n$
for $i = 1$ to n **do**
    $t_i \sim T$
    $u_i \leftarrow \mathcal{N}(0_N, I_{N\cdot N})$
    $g \leftarrow g + F(t_i(x+\sigma\cdot u_i))_y \cdot u_i$
    $g \leftarrow g - F(t_i(x-\sigma\cdot u_i))_y \cdot u_i$
end for
return $\frac{1}{2n\sigma}g$

We use the same class of transformation functions listed in Section 3.3.3 for the distribution T. Algorithm 2 details our implementation for estimating gradients for crafting robust adversarial examples. We follow the same update rule given by Equation 3.4 to generate adversarial frames. We iterate until a given a number of maximum iterations or until the attack is successful under the sampled set of transformation functions.

3.3.6 Universal attack

While the transferability of adversarial perturbations poses a practical threat to Deepfake detectors in production, creating an adversarial video requires significant technical expertise in adversarial machine learning — the attacker needs to solve an optimization problem for each frame of the video to fool the detector.

To ease the process of fooling Deepfake detectors, we aim to design more accessible adversarial attacks that can be easily shared amongst attackers. Past works [115, 12, 119] have shown the existence of universal adversarial perturbations that can fool classification models in various input domains. We aim to find a single universal adversarial perturbation which when added across all frames of any video, will cause the victim Deepfake Detector to classify the video to a target label.

That is, we aim to find a targeted universal perturbation δ such that:

$$\begin{aligned} &C(x+\delta) = y \quad s.t \quad ||\delta||_\infty < \varepsilon \\ &\text{for "most" } x \text{ in our dataset} \end{aligned} \tag{3.5}$$

where y is the target class. We train separate perturbations for Real and Fake target labels. In order to ensure robustness to differences across detection methods, we incorporate the transformation functions described in Section 3.3.3. We train the universal adversarial perturbation on a dataset of videos that are labelled opposite from our target label. On this dataset of videos, we aim to maximize the log-likelihood of predicting our target label y. Additionally to ensure the imperceptibility of the adversarial perturbation we penalize the L_2 distortion of the perturbation by adding a regularization term in our objective. Thus, our final objective to train a universal perturbation for a target label y is as follows:

$$\begin{aligned} &\textit{Minimize} \sum_{x \text{ in } D} \mathbb{E}_{t\sim T}[L(F(t(x+\delta)), y)] + c||\delta||_2 \\ &\textit{such that} \quad ||\delta||_\infty < \varepsilon \end{aligned} \tag{3.6}$$

Here, L is the cross-entropy loss between the predictions and our target label, c is a hyper-parameter to control the regularization loss and x is an input frame of a video from our dataset D. Similar to Equation 3.3, we estimate the above expectation using n samples as follows:

$$\mathbb{E}_{t\sim T}[L(F(t(x+\delta)),y)] = \frac{1}{n}\sum_{t_i\sim T}[L(F(t_i(x+\delta)),y)] \tag{3.7}$$

To ensure the constraint $||\delta||_\infty < \varepsilon$, we express δ as follows:

$$\delta = \varepsilon \cdot tanh(p)$$

where p is a trainable unconstrained parameter having the same dimensions as δ. We fix the size of the perturbation vector p to be $3 \times 256 \times 256$ in our experiments, but resize the perturbation using bilinear interpolation to match the size of our input x. We iteratively optimize the objective given by Equation 3.6 using gradient descent. In our experiments, we find that targeting certain Deepfake detectors not only results in input-agnostic universal perturbations but also model-agnostic universal perturbations.

3.4 Experimental Setup

We perform adversarial attacks on the FaceForensics++ [139] and the DFDC datasets [46] and choose the best performing models on these datasets as the victim models. We first craft adversarial videos for the FaceForensics++ dataset and target the XceptionNet and MesoNet models which are the best reported architectures reported in the paper [139] introducing this dataset (Section 3.5.1). We use these two models as a test-bed to study the robustness of our attacks to video compression and demonstrate the using our robust attack helps significantly improve attack performance on compressed videos. Next we conduct the transferable and universal attack experiments on the DFDC dataset. We choose the models from top three winning entries in the DFDC Kaggle competition as the victim models for these experiments

(Section 3.5.2, 3.5.3). Finally, we evaluate our attacks on a sequence based 3D CNN model to demonstrate that adversarial examples are a threat to not only frame by frame detectors but also sequence based models (Section 3.5.4).

3.4.1 Dataset and Models

On the FaceForensics++ dataset, XceptionNet [32] and MesoNet [2] CNN classifiers have been reported to achieve the best performance in the paper introducing the dataset [139]. For these two models, we perform our attack on the test set of the FaceForensics++ Dataset [139], consisting of manipulated videos from the four methods described in Section 3.1. We construct adversarially modified fake videos on the FaceForensics++ test set, which contains 70 videos (total 29,764 frames) from each of the four manipulation techniques. For simplicity, our experiments are performed on high quality (HQ) videos, which apply a light compression on raw videos. The accuracy of the detector models for detecting facially manipulated videos on this test set is reported in Table 3.1.

Table 3.1. Accuracy of Deepfake detectors on the FaceForensics++ HQ Dataset as reported in [139]. The results are for the entire high-quality compressed test set of Deepfakes.

	DF	**F2F**	**FS**	**NT**
XceptionNet [139] Acc %	97.49	97.69	96.79	92.19
MesoNet [139] Acc %	89.55	88.6	81.24	76.62

For the DFDC dataset, we choose the top three winners of the challenge, which was hosted by Facebook on the Kaggle website. The top two winning entries of the challenge rely solely on face detection models and per-frame CNN classifiers similar to the best performing models on the FaceForensics++ dataset. The third place winner of the challenge uses a combination of per-frame classifiers and a 3D CNN based sequence model. Table 3.2 lists the Deepfake detection methods studied in this work along with their respective CNN architectures used for classification and face detection. We use the DFDC dataset and these top three winning models as the test bed for evaluating the transferability of our attacks across different models. In our

Table 3.2. Different Deepfake detection systems studied in our work with their respective classification models, face detection models and detection AUC scores on the DFDC test set.

Model	*Team Name*	*Classifier*	*Face detection*	*AUC*
EN-B7 Selim [143]	Selim	EfficientNet B7 [159]	MTCNN [186]	0.717
XN WM [65]	Team WM	XceptionNet [32]	RetinaFace [43]	0.724
EN-B3 WM [65]	Team WM	EfficientNet B3 [159]	RetinaFace [43]	0.724
EN-B7 NLab [40]	NTech Lab	EfficientNet B7 [159]	DSFD [97]	0.717

transferability experiments we use the terms *victim model* and *test model* and define them as:

- *Victim model:* The detection model that the attack/adversarial perturbation is trained on, in the complete-knowledge (white-box) attack scenario.
- *Test model:* The model on which we evaluate the attack. This can be the same as the victim model (white-box) or an unseen detection model (black-box).

We craft adversarial videos for the first 100 Fake and 100 Real videos in the public DFDC validation set [46]. These videos contain a total of 30,300 frames. The videos are recorded in various lighting and background conditions and include people with different skin-tones.

3.4.2 Evaluation Metrics

Once the adversarial frames are generated, we combine them and save the adversarial videos in the following formats:

- *Uncompressed (Raw):* Video is stored as a sequence of uncompressed images.
- *Compressed (MJPEG):* Video is saved as a sequence of JPEG compressed frames.
- *Compressed (H.264):* Video is saved in the commonly used mp4 format that applies temporal compression across frames.

We conduct our primary evaluation on the *Raw* and *MJPEG*. We also study the effectiveness of our white box robust attack using different compression levels in the *H264* codec. We report the following metrics for evaluating our attacks:

- **Success Rate (SR)**: The percentage of frames in the adversarial videos that get classified to our target label. We report: **SR-U**- Attack success rate on uncompressed adversarial videos saved in Raw format; and **SR-C**- Attack success rate on compressed adversarial videos saved in MJPEG format.

- **Accuracy**: The percentage of frames in videos that get classified to their original label by the detector. We report **Acc-C**- accuracy of the detector on compressed adversarial videos.

- **Mean distortion ($\mathbf{L_\infty}$)**: The average L_∞ distortion between the adversarial and original frames. The pixel values are scaled in the range [0,1], so changing a pixel from full-on to full-off in a grayscale image would result in L_∞ distortion of 1 (not 255).

3.5 Results

3.5.1 Evaluation on FaceForensics++ dataset

Simple white-box attack

To craft adversarial examples in the white-box setting, in our attack pipeline, we implement differentiable image pre-processing (resizing and normalization) layers for the CNN. This allows us to backpropagate gradients all the way to the cropped face in-order to generate the adversarial image that can be placed back in the frame. We set the maximum number of iterations to 100, learning rate α to $1/255$ and max L_∞ constraint ε to $16/255$ for both our attack methods described in Sections 3.3.2 and 3.3.3.

Table 3.3 shows the results of the white-box attack (Section 3.3.2). We are able to generate adversarial videos with an average success rate of 99.85% for fooling XceptionNet and 98.15% for MesoNet when adversarial videos are saved in the Raw format. However, the

Table 3.3. Evaluation of various attacks on the two models XceptionNet and MesoNet on the FaceForensics++ dataset. We report the average L_∞ distortion between the adversarial and original frames and the attack success rate on uncompressed (SR-U) and compressed (SR-C) videos.

		XceptionNet				MesoNet			
Attack	Dataset	$\mathbf{L}_\infty$	SR - U	SR - C	Acc-C %	$\mathbf{L}_\infty$	SR - U	SR - C	Acc-C %
Simple White-box (Section 3.3.2)	DF	0.004	99.67	43.11	56.89	0.006	97.30	92.27	7.73
	F2F	0.004	99.85	52.50	47.50	0.007	98.94	96.30	4.70
	FS	0.004	100.00	43.13	56.87	0.009	97.12	86.10	13.90
	NT	0.004	99.89	95.10	4.90	0.007	99.22	96.20	3.80
	All	0.004	99.85	58.46	41.54	0.007	98.15	92.72	7.53
Robust and Transferable (Section 3.3.3)	DF	0.016	99.56	98.71	1.29	0.030	99.94	99.85	0.15
	F2F	0.013	100.00	99.00	1.00	0.020	99.71	99.67	0.33
	FS	0.013	100.00	95.33	4.67	0.026	99.02	98.50	1.50
	NT	0.011	100.00	99.89	0.11	0.025	99.99	99.98	0.02
	All	0.013	99.89	98.23	1.77	0.025	99.67	99.50	0.50
Query based Black-box (Section 3.3.4)	DF	0.055	89.72	55.64	44.36	0.062	96.05	93.33	6.67
	F2F	0.055	92.56	81.40	18.60	0.0627	84.08	77.68	22.32
	FS	0.045	96.77	23.50	76.50	0.0627	77.55	62.44	37.56
	NT	0.024	99.86	94.23	5.77	0.0627	85.98	79.25	20.75
	All	0.045	94.73	63.69	36.31	0.0626	85.92	78.18	21.83
Query based Robust Black-box (Section 3.3.5)	DF	0.060	88.47	79.18	20.82	0.047	96.19	93.80	93.80
	F2F	0.058	97.68	94.42	5.58	0.054	84.14	77.50	77.50
	FS	0.052	98.97	63.26	36.74	0.061	77.34	61.77	61.77
	NT	0.018	99.65	98.91	1.09	0.053	88.05	80.27	80.27
	All	0.047	96.19	83.94	16.06	0.053	86.43	78.33	78.33

attack average success rate drops to 58.46% for XceptionNet and 92.72% for MesoNet when MJPEG compression is used. This result is coherent with past works [51, 39, 64] that employ JPEG compression and image transformations to defend against adversarial examples.

Robust attack

For our robust white box attack, we sample 12 transformation functions from the distribution T for estimating the gradient in each iteration. This includes three functions from each of the four transformations listed in Section 3.3.3. Table 3.4 shows the search distribution for different hyper-parameters of the transformation functions.

Table 3.4. Search distribution of hyper-parameters of different transformations used for our Robust White box attack. During training, we sample three functions from each of the transforms to estimate the gradient of our expectation over transforms.

Transform	Hyper-parameter search distribution
Gaussian Blur	Kernel $k(d,d,\sigma)$, $d \sim \mathscr{U}[3,7]$, $\sigma \sim \mathscr{U}[5,10]$
Gaussian Noise	$\sigma \sim \mathscr{U}[0.01,0.02]$
Translation	$d_x \sim \mathscr{U}[-20,20]$, $d_y \sim \mathscr{U}[-20,20]$
Down-sizing & Up-sizing	Scaling factor $r \sim \mathscr{U}[2,5]$

Table 3.3 shows the results of our robust white-box attack. It can be seen that robust white-box is effective in both Raw and MJPEG formats. The average distortion between original and adversarial frames in the robust attack is higher as compared to the non-robust white-box attack. We achieve an average success rate (SR-C) of 98.07% and 99.83% for XceptionNet and MesoNet respectively in the compressed video format.

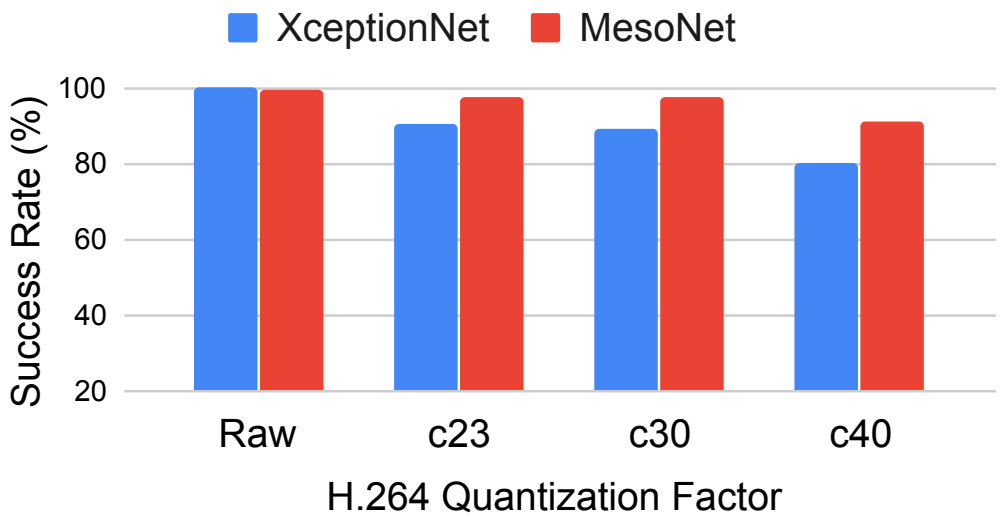


Figure 3.4. Attack success rate vs Quantization factor used for compression in H264 codec for robust white box attack.

We also study the effectiveness of our robust white box attack under different levels of compression in the H.264 format which is widely used for sharing videos over the internet. Figure 3.4 shows the average success rate of our attack across all datasets for different quantization

parameter c used for saving the video in H.264 format. The higher the quantization factor, the higher the compression level. In [139], fake videos are saved in HQ and LQ formats which use $c = 23$ and $c = 40$ respectively. It can be seen that even at very high compression levels ($c = 40$), our attack is able to achieve 80.39% and 90.50% attack success rates for XceptionNet and MesoNet respectively, without any additional hyper-parameter tuning for this experiment. Examples of adversarial frames are showin in Figure 3.5 and video examples can be found on the website linked in the footnote [2].

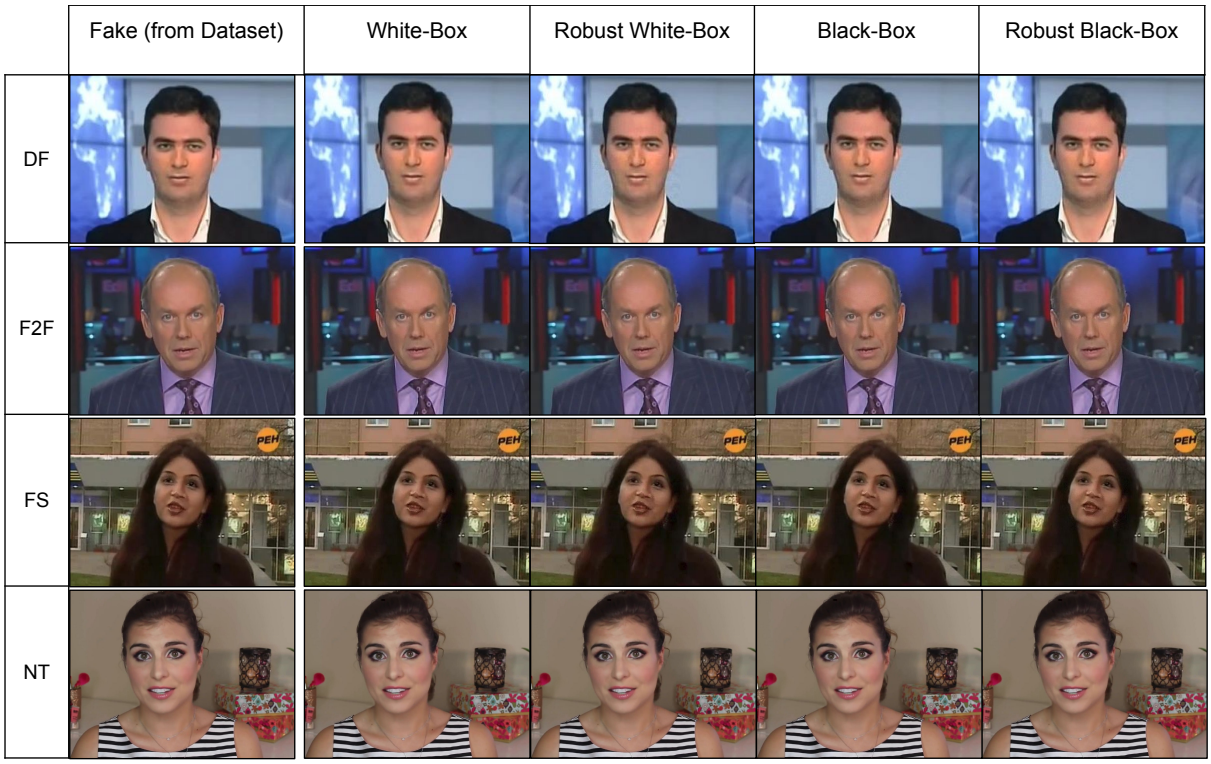


Figure 3.5. Randomly selected frames of Adversarial Deepfakes from successful attacks. Video examples are linked in the footnote.

3.5.2 Transferability of adversarial attacks

We evaluate the transferability of adversarial perturbations across different detectors trained on the DFDC dataset. We train adversarial videos targeting a given victim model and test

[2]https://adversarialdeepfakes.github.io/

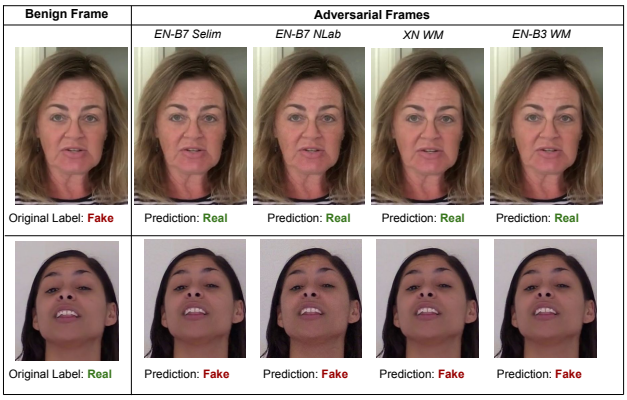
the videos against different test models. For our simple whitebox attack, while we achieve 100% attack success rate for the same test model as the victim model, the attack success rate drops significantly on alternate models. EfficientNet-B7 by NTech Lab requires the highest amount of adversarial perturbation under the L_∞ metric as compared to other methods in this study. We find that perturbations trained to fool EfficientNet-B7 by Team NTech Lab result in the most transferable attacks as indicated by the higher success rates on other test models. This suggests that *EN-B7 NLab* is relatively more robust to adversarial perturbations in comparison to the other models used in this study (also indicated by higher L_∞ perturbation required to fool *EN-B7 NLab*).

To improve the transferability of adversarial examples across different methods, we perform our robust transfer attack described in Section 3.3.3 and evaluate the adversarial videos against unseen detection methods in a black-box setting. The hyper-parameters of the transformation functions used for the attack have been provided in Table 3.4. All other attack hyper-parameters are kept the same as our simple white-box attack.

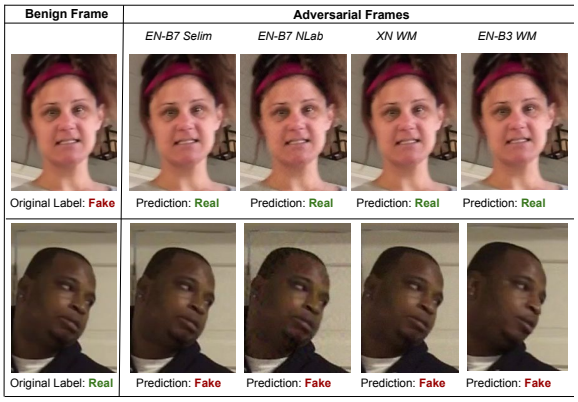
As indicated by the results in Table 3.5, we are able to significantly improve the transferability of adversarial perturbations across different detection methods as compared to our simple white-box attack. The adversarial perturbations are most transferable across models with the same architecture. For example, we are able to achieve high cross-transferability between *EN-B7 Selim* vs *EN-B7 NLab*. Similar to our observation in the previous section, attacking *EN-B7 NLab* results in the most transferable adversarial attacks - we are able to achieve at least 72% success rate across all other detection methods when attacking *EN-B7 NLab*. Sample images for these attacks are presented in Figure 3.6.

3.5.3 Universal attacks

To create more accessible attacks, we train a universal adversarial perturbation using the procedure described in Section 3.3.6. We set the L_2 regularization term $c = 0.01$ and use the Adam optimizer with a learning rate of 0.001. For our initial experiments, we set the L_∞



Simple White-box attack examples on DFDC detectors



Robust and Transferable attack examples on DFDC detectors

Figure 3.6. Randomly selected frames of adversarial videos from attacks on the DFDC detectors.

Table 3.5. Attack success rates (SR-U) of the *white-box* (Section 3.3.2) and *robust and transferable attacks* (Section 3.3.3) on different victim models and their transferability to seen and unseen detectors (test models).

			Test Model			
	Victim Model	L_∞	EN-B7 Selim	EN-B7 NLab	XN WM	EN-B3 WM
Simple White-box (Section 3.3.2)	EN-B7 Selim	0.007	100.0 %	59.5 %	57.0 %	38.5 %
	EN-B7 NLab	0.013	94.0 %	100.0 %	66.5 %	49.5 %
	XN WM	0.006	13.0 %	12.5 %	100.0 %	12.0 %
	EN-B3 WM	0.005	21.0 %	15.5 %	22.0 %	100.0 %
Robust and Transferable (Section 3.3.3)	EN-B7 Selim	0.010	100.0 %	89.0 %	72.5 %	62.0 %
	EN-B7 NLab	0.018	99.0 %	100.0 %	72.0 %	76.5 %
	XN WM	0.018	49.0 %	33.5 %	100.0 %	46.0 %
	EN-B3 WM	0.008	46.5 %	35.0 %	47.5 %	100.0 %

threshold $\varepsilon = 40/255$ for all victim models. Since the goal of finding a single input-agnostic perturbation is more challenging than finding one perturbation per video frame, a higher amount of distortion is required for a successful attack as compared to the per-frame attacks described earlier. We train the universal perturbation on a dataset of 100 videos from the DFDC train set which are separate from our evaluation dataset. We train the perturbation using a batch size of 8 for $10,000$ iterations.

We target one victim model at a time and test the transferability of the universal perturbation on seen and unseen detectors. Table 3.6 presents the results of performing the universal attack on different victim models at $\varepsilon = 40/255 = 0.156$. We are able to achieve 100% attack success rate on the same test model as the victim model using a single perturbation across

Table 3.6. Attack success rates (SR-U) of the universal attacks (Section 3.3.6) on different victim models and their transferability to unseen detectors (test models).

		Test Model			
Victim Model	L_∞	EN-B7 Selim	EN-B7 NLab	XN WM	EN-B3 WM
EN-B7 Selim	0.156	100.0%	94.5%	65.0%	69.0%
EN-B7 NLab	0.156	94.5%	100.0%	75.0%	81.5%
XN WM	0.156	77.5%	61.0%	100.0%	20.0%
EN-B3 WM	0.156	66.5%	50.5%	60.0%	100.0%

all frames and videos of the same label. Also, the universal perturbation is transferable to a significant extent across different models which poses an extremely practical threat to Deepfake detectors in production. Attacking *EN-B7 NLab* results in the most transferable perturbations where we are able to achieve at least a 75% success rate across all unseen detectors.

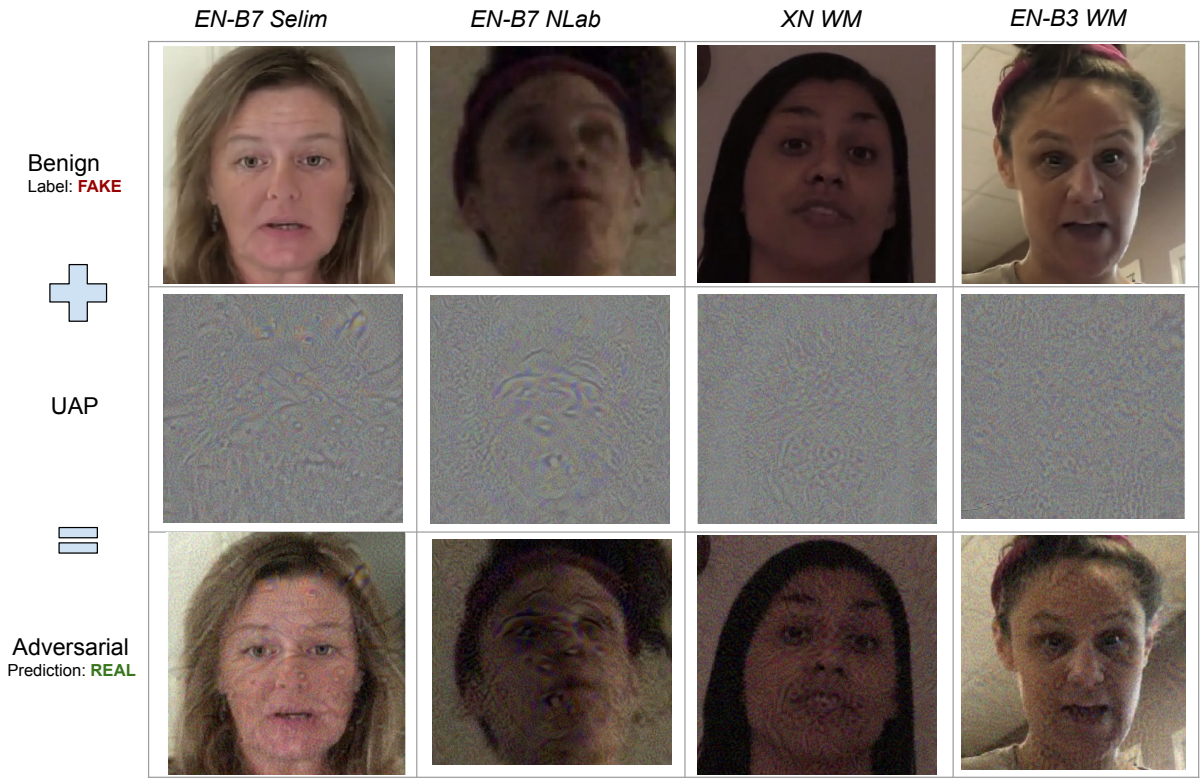


Figure 3.7. Visualization of universal adversarial perturbations trained on different Deepfake detection models at $\varepsilon = 0.156$.

Visually, the universal perturbations at $\varepsilon = 0.156$ are more perceptible than our per-frame

attacks discussed in the sections above. Figure 3.7 shows examples of universal adversarial perturbations trained on different Deepfake detectors and the resulting adversarial images obtained after adding the perturbation to the face-crop of the benign frame.

We perform an additional experiment to study the effectiveness of universal adversarial perturbations at different magnitudes of added perturbations. We choose *EN-B7 NLab* as the victim model and perform our universal attack at different values of ε. The attack success rates across different models are shown in Figure 3.8. Figure 3.8 also shows what a perturbed image looks like at different values of ε. At $\varepsilon < 0.1$, the perturbation is fairly imperceptible but can still achieve high success rates on various test models.

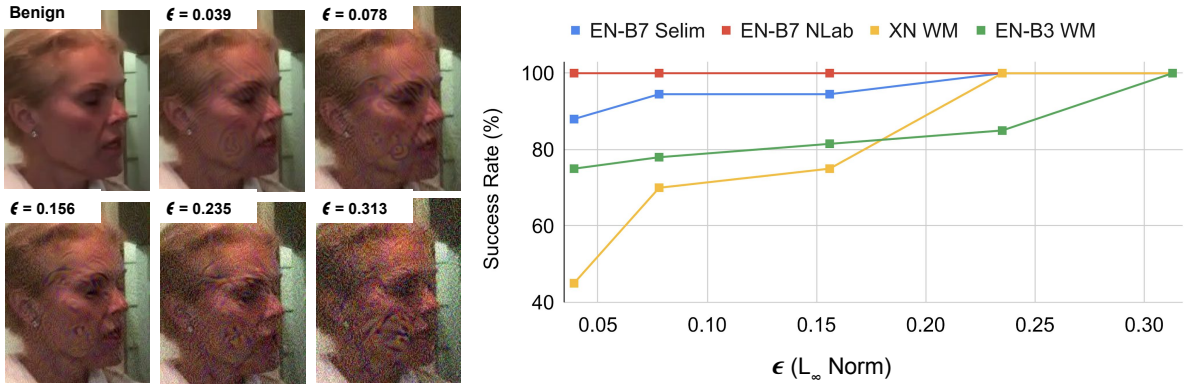


Figure 3.8. *Left:* Visualization of the perturbed images using different magnitudes (ε) of universal adversarial perturbations trained on *EN-B7 NLab*. *Right:* Attack success rates of the universal attacks (Section 3.3.6) on different victim models and their transferability to unseen detectors (test models).

3.5.4 Evaluation on Sequence Based Detector

We consider the 3D CNN based detector described in Section 3.2. The detector performs 3D convolution on a sequence of face-crops from 7 consecutive frames. We perform our attacks on the pre-trained model checkpoint (trained on the DFDC [46] train set) released by the NTechLab team [40]. We evaluate our attacks on the Deepfake videos from the DFDC public validation set which contains 200 Fake videos. We report the accuracy of the detector on the 7-frame sequences from this test set in the first row of Table 3.7.

Table 3.7. Evaluation of different attacks on a sequence based detector on the DFDC validation dataset. The first row indicates the performance of the classifier on benign (non adversarial) videos.

	3D CNN Sequence Model			
Attack Type	**L_∞**	**SR - U**	**SR - C**	**Acc. - C%**
None	-	-	-	91.74
Simple White-box (Section 3.3.2)	0.037	100.00	77.67	22.33
Robust and Transferable (Section 3.3.3)	0.059	100.00	100.00	0.00
Query based Black-box (Section 3.3.4)	0.061	87.99	24.43	75.57
Query based robust Black-box (Section 3.3.4)	0.062	88.21	51.02	48.98

Similar to our attacks on frame-by-frame detectors, in the white-box setting we back-propagate the loss through the entire model to obtain gradients with respect to the input frames for crafting the adversarial frames. While both white-box and robust white-box attacks achieve 100% success rate on uncompressed videos, the robust white-box attack performs significantly better on the compressed videos and is able to completely fool the detector. As compared to frame-by-frame detectors, a higher magnitude of perturbation is required to fool this sequence model in both the white-box attacks. In the black-box attack setting, while we achieve similar attack success rates on uncompressed videos as the frame-by-frame detectors, the attack success rate drops after compression. The robust black-box attack helps improve robustness of adversarial perturbations to compression as observed by higher success rates on compressed videos (51.02% vs 24.43% SR-C).

3.6 Conclusion

In this chapter, I described the current best-performing Deepfake classifiers and studied their vulnerability to adversarial examples. We consider both per-frame and sequence-based Deepfake detection models and demonstrate that they can be bypassed under various attack settings and attacker capabilities. We first design an attack pipeline to bypass Deepfake detectors in a white-box attack setting and propose techniques to increase the robustness of such attacks

to video compression codecs. Next, we demonstrate that adversarial videos crafted using our robust attacks can fool alternate models to a significant extent thereby posing a real-world threat in a black-box attack setting. Finally, we demonstrate the existence of universal adversarial perturbations which pose a more practical threat since they can be easily shared amongst attackers and applied to any video in real-time. In the upcoming chapter, I discuss a semi-fragile watermarking framework as a proactive media authentication method to overcome the limitations of Deepfake classifiers.

3.7 Acknowledgements

Chapter 3 contains material found in the following two papers. (1) *Adversarial Deepfakes: Evaluating Vulnerability of Deepfake Detectors to Adversarial Examples*. 2021. Hussain, Shehzeen; Neekhara, Paarth; Jere, Malhar; Koushanfar, Farinaz; McAuley, Julian. Winter Conference on Applications of Computer Vision 2021. (2) *Exposing Vulnerabilities of Deepfake Detection Systems with Robust Attacks*. 2023. Hussain, Shehzeen; Neekhara, Paarth; Dolhansky, Brian; Bitton, Joanna; Canton, Cristian; McAuley, Julian; Koushanfar, Farinaz. ACM Journal on Digital Threats: Research and Practice, Vol 3, 2022. The dissertation author and Shehzeen Hussain made equal contributions to this work.

Chapter 4

Media authentication using Proactive Watermarking

Media authentication, despite having been a long-term challenge, has become even more difficult with the advent of deep learning based generative models. As discussed in the previous chapters, Deep Neural Network (DNN) based generative models, have enabled the creation of high-quality synthetic media in various domains. Such techniques can be used to easily manipulate real images, videos and audio to fuel misinformation, tamper sensitive documents, defame individuals and reduce trust in social media platforms [111]. Media authentication is crucial in ensuring the accuracy of news and maintaining public trust to safeguard against the potential misuse of generative models. Media authentication also plays a crucial role in law enforcement, where videos and images are often used as evidence. Recent methods to detect fake media rely on DNN based classification systems [139, 45]. As discussed in the previous chapter, there are certain limitations in classification based fake media detectors: 1) The current best-performing detectors for synthetic media can be easily bypassed by attackers using adversarial examples. 2) Classifiers trained in a supervised manner on existing media synthesis techniques cannot be reliably secure against black-box generation methods.

As an alternate solution to fake media detection, proactively embedding a secret verifiable message into images and videos at the time of their capture from a device can establish the provenance of authentic images and videos and circumvent the limitations of classifiers for

synthetic media. Prior work has explored digital image watermarking and deep learning based steganography techniques [37, 53, 160, 191, 109] to hide secret messages in image pixels. However, these methods are either fragile to basic image processing operations such as compression and color adjustments or overly robust to the point that the secret can be recovered even after occluding major portions of the embedded image [160]. Moreover, past neural watermarking frameworks are not designed to be robust to common video compression codecs that apply temporal compression along with per-frame spatial compression. For solving the challenge of media authentication, the watermarking framework should have the following desirable properties: 1) The watermark data should be recoverable if the image/video undergoes benign transformations such as compression or minor adjustments. 2) The watermark recovery should break if the image/video has been maliciously manipulated e.g. replacing the face, occluding/replacing significant portions of the image 3) The watermark should be visually imperceptible.

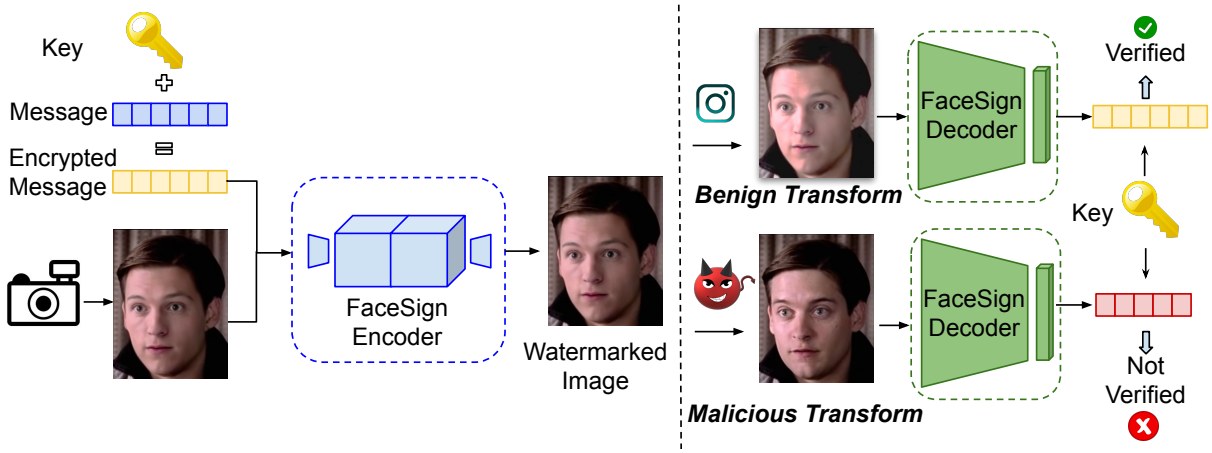


Figure 4.1. Overview of *FaceSigns* Watermarking framework: The encoder network embeds a secret encrypted message into a given image as an imperceptible watermark that is designed to be robust against benign transformations but fragile towards malicious manipulations.

In this chapter, I describe *FaceSigns*, a semi-fragile image and video watermarking framework that we developed to embed a recoverable message as an imperceptible perturbation in the image pixels. The watermark can contain a secret message or device-specific codes which can be used for authenticating images and videos. The desirable property of the watermark is that

it should break if a malicious manipulation such as occlusion, face-swapping or content manipulation is applied to the image/video, but it should be robust against harmless transformations such as image compression, video compression, color and lighting adjustments which are commonly applied on pictures and videos before uploading them to online sharing platforms. To achieve this goal, we develop an encoder-decoder based training framework that encourages message recovery under benign transformations and discourages message recovery if the watermark has been spatially tampered in certain parts of the image. In contrast to hand-designed pipelines used in previous work for semi-fragile watermarking [100, 67, 179, 132, 16], *FaceSigns* is trained end-to-end and learns to be robust to a wide range of real-world digital image processing operations such as social media filters and compression techniques, while being fragile to various Deepfake tampering techniques.

4.1 Background: Digital Watermarking

Digital watermarking [37] similar to steganography [53], is the task of embedding information into an image in a visually imperceptible manner. These techniques broadly seek to generate three different types of watermarks: fragile [44, 16], robust [36, 129, 17, 146, 191, 128, 13] and semi-fragile [100, 155, 181] watermarks. Fragile and semi-fragile watermarks are primarily used to certify the integrity and authenticity of image data. Fragile watermarks are used to achieve accurate authentication of digital media, where even a one-bit change to an image will lead it to fail the certification system. In contrast, *robust* watermarks aim to be recoverable under several image manipulations, in order to allow media producers to assert ownership over their content even if the video is redistributed and modified. Semi-fragile watermarks combine the advantages of both robust and fragile watermarks, and are mainly used for fuzzy authentication of digital images and identification of image tampering [181]. The use of semi-fragile watermarks is justified by the fact that images and videos are generally transmitted and stored in a compressed form, which should not break the watermark. However when the image gets tampered, the

watermark should also get damaged, indicating image tampering.

Several past works have proposed hand-engineered pipelines to embed semi-fragile watermark information in the spatial and frequency (transform) domain of images and videos. In the spatial domain, the pixels of digital images are processed directly using block-based embedding [16] and least significant bits modification [176, 179] to embed watermarks. In the frequency domain, the watermark can be embedded by modifying the coefficients produced with transformations such as the Discrete Cosine Transform (DCT) [132, 67, 13] and Discrete Wavelet Transform [96, 15, 145]. However, we demonstrate in our experiments the major limitations of traditional approaches lies in higher visibility of the embedded watermarks, increased distortions in generated images and low robustness to compression techniques like JPEG transforms. Moreover, these works have not been designed to be fragile against Deepfake manipulations.

More recently, CNNs have been used to provide an end-to-end solution to the watermarking problem. They replace hand-crafted hiding procedures with neural network encoding [8, 66, 191, 188, 109, 160]. Notably, both StegaStamp [160] and HiDDeN [191] propose frameworks to embed robust watermarks that can hide and transmit data in a way that is robust to various real-world transformations. All of these works focus on generating robust watermarks, with the goal of ensuring robustness and recovery of the embedded secret information under various physical and digital image distortions. We empirically demonstrate that these techniques are unable to generate semi-fragile watermarks and are therefore not suitable for identifying tampered media such as Deepfakes.

4.2 Methodology

In this section, we describe the training methodology for the image watermarking framework FaceSigns that can withstand a range of benign image and video alterations, while remaining fragile to malicious ones. It is also crucial that the watermark is unnoticeable, allowing the

devices to retain solely the watermarked images without exposing the original image to the end-user. The selection of benign and malicious transformations is dependent on the media authentication system's application and can be adjusted as needed. For instance, in situations where document verification is the primary objective, it may be preferable to limit benign transformations to compression only, whereas for social media platforms, creative image filtering may be permitted. We are proposing a flexible framework that can be adapted to accommodate any benign and malicious transformations.

Our system consists of three main components: an encoder network E_α, a decoder network D_β and an adversarial discriminator network A_γ where α, β and γ are learnable parameters. The encoder network E takes as input an image x and a bit string $s \in \{0,1\}^L$ of length L, and produces an encoded (watermarked) image x_w. That is, $x_w = E(x,s)$. The watermarked image then goes through two image transformation functions — one sampled from a set of benign transformations ($g_b \sim G_b$) and the other sampled from a set of malicious transformations ($g_m \sim G_m$) to produce a benign image $x_b = g_b(x_w)$ and a malicious image $x_m = g_m(x_w)$. The benign and malicious watermarked images are then fed to the decoder network which predicts the messages $s_b = D(x_b)$ and $s_m = D(x_m)$ respectively.

For optimizing secret retrieval during training, we use the L_1 distortion between the predicted and ground-truth bit strings. The decoder is encouraged to be robust to benign transformations by minimizing the message distortion $L_1(s,s_b)$; and fragile for malicious manipulations by maximizing the error $L_1(s,s_m)$. Therefore the secret retrieval error for an image $L_M(x)$ is obtained as follows:

$$L_M(x) = L_1(s,s_b) - L_1(s,s_m) \tag{4.1}$$

The watermarked image is encouraged to look visually similar to the original image by optimizing three image distortion metrics: L_1, L_2 and L_{pips} [187] distortions. Additionally, we use an adversarial loss $L_G(x_w) = \log(1 - A(x_w))$ from the discriminator which is trained simultaneously to distinguish original images from watermarked images. That is, our image

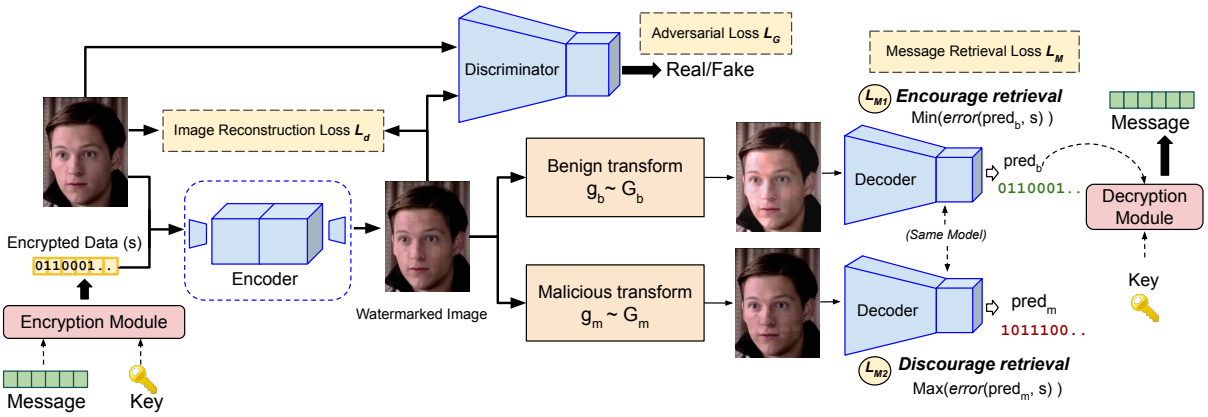


Figure 4.2. Model overview: The encoder and decoder networks are trained by encouraging watermark imperceptibility and message retrieval from watermarked images that have undergone benign transformations and discouraging retrieval from maliciously transformed watermarked images.

reconstruction loss L_{img} is obtained as follows:

$$\begin{aligned} L_d(x, x_w) &= L_1(x, x_w) + L_2(x, x_w) + c_p L_{pips}(x, x_w) \\ L_{img}(x, x_w) &= L_d(x, x_w) + c_g L_G(x_w) \end{aligned} \tag{4.2}$$

Therefore, the parameters α, β of the encoder and decoder network are trained using mini-batch gradient descent to optimize the following loss over a distribution of input messages and images:

$$\mathbb{E}_{x,s,g_b,g_m}[L_{img}(x, x_w) + c_M L_M(x)] \tag{4.3}$$

The discriminator parameters γ are trained to distinguish original images x from watermarked images x_w as follows:

$$\mathbb{E}_{x,s}[\log(1 - A(x)) + \log(A(x_w))] \tag{4.4}$$

In the above equations, c_p, c_g, c_M are scalar coefficients for the respective loss terms. We use the following values for our loss coefficients: $c_p = 1$, $c_g = 0.1$, $c_m = 1$. We use Adam optimizer during training with a learning rate of $2e-4$.

4.2.1 Message encoding

The encoder network accepts watermarking data as a bit string s of length L. This watermarking data can contain information about the device that captured the image or a secret message that can be used to authenticate the image. To prevent adversaries (who have gained white-box access to the encoder network) from encoding a target message, we can encrypt the message using symmetric or asymmetric encryption algorithms or hashing. In our experiments, we embed encrypted messages of size 128 bits which allows the network to encode 2^{128} unique messages. We discuss the possible threats and defenses to our watermarking framework in Section 4.4.

4.2.2 Network Architectures

Our encoder and decoder networks are based on the U-Net CNN architecture [138, 73, 160] and operate on 256×256 images. The encrypted message s, which is an L length bit string, is first projected to a tensor s_{Proj} of size 96×96 using a trainable fully connected layer; then resized to 256×256 using bilinear interpolation and finally added as the fourth channel to the original RGB image to be fed as an input to the encoder network. The encoder U-Net contains 8 downsampling and 8 upsampling layers. We modify the original U-Net architecture and replace the transposed convolution in the upsampling layers with convolutions followed by nearest-neighbour upsampling as per the recommendations given by [122]. In our preliminary experiments, we found this change to significantly improve the image quality and training speed of our framework. The downsampling and upsampling layers have skip-connections between the corresponding layers with same output size. The decoder network also follows the U-Net architecture similar to our encoder network. The decoder U-Net first outputs a 256×256 intermediate output, which is downsized to 96×96 using bilinear down-sampling to produce $s_{ProjDecoded}$ and then projected to a vector of size L using a fully connected layer followed by a sigmoid layer to scale values between 0 and 1. We use batch normalization layers in the encoder

network and instance normalization layers in the decoder network.

4.2.3 Transformation functions

To achieve selective fragility and robustness of the watermark, it is crucial to carefully select benign and malicious transformation functions. Although we can only employ a restricted range of image transformations for training purposes, the real-world scenarios involve an extensive set of possible benign and malicious transformations, which cannot be exhaustively listed. Our experiments (described in Section 4.3.3) demonstrate that by incorporating the transformation functions described below, we can effectively adapt to unforeseen benign and malicious transformations that are commonly employed on various social media platforms.

Benign Transforms

To approximate standard image processing distortions, we apply a diverse set of differentiable benign image transformations (G_b) to our watermarked images during training:

1. **Gaussian Blur:** We convolve the original image with a Gaussian kernel k. This transform is given by $t(x) = k * x$ where $*$ is the convolution operator. We use kernel sizes ranging from $k = 3$ to $k = 7$

2. **JPEG compression:** Digital images are usually stored in a lossy format such as JPEG. We approximate JPEG compression with the differentiable JPEG function proposed in [148]. During training, we apply JPEG compression with quality 40, 60 and 80.

3. **Saturation adjustments:** To account for various color adjustments from social media filters, we randomly linearly interpolate between the original (full RGB) image and its grayscale equivalent.

4. **Contrast adjustments:** We linearly rescale the image histogram using a contrast factor $\sim \mathscr{U}[0.5, 1.5]$

5. **Downsizing and Upsizing:** The image is first downsized by a factor *scale* and then up-sampled by the same factor using bilinear upsampling. We use $scale \sim \mathscr{U}[2,5]$

6. **Video Compression:** Simulating video compression distortions during training is more challenging because common video compression codecs such as MPEG4, H264 cannot be easily implemented using differentiable functions. Such codecs not only compress each frame of a given video but also apply temporal compression across the time-steps for a more optimised compression. Since video compression is applied to almost all videos uploaded on the internet, it is essential to ensure the robustness to these codecs to make the watermark suitable for videos. To this end, we propose the first technique to ensure robustness of the generated watermark to a benign non-differentiable video transform g_b:

    - When training the watermarking framework on videos, each mini-batch of images x, corresponds to consecutive frames of a single video.
    - We obtain watermarked frames x_w by embedding unique signatures into each frame using the encoder network. Next, we detach x_w from the computational graph, extract each frame and write the frames into a video file. The video file is then compressed into H264 codec using FFMPEG [1] at a quantization factor from the interval $[5,25]$.
    - Next, we read each frame of the compressed file and stack them together to obtain the transformed image batch $g_b(x_w)$ which is then reinserted in the computational graph to be fed as input to the decoder.
    - During the backward pass, we use the straight-through estimator [14] to estimate the gradient across the transformation function g_b. That is:

$$\nabla_{x_w} L_M(g_b(x_w))|_{x_w=\hat{x_w}} \approx \nabla_{x_w} L_M(x_w)|_{x_w=g_b(\hat{x_w})}. \tag{4.5}$$

    where $L_M(x_w)$ indicates the message recovery loss from the decoder for an input x_w.

[1] https://ffmpeg.org/

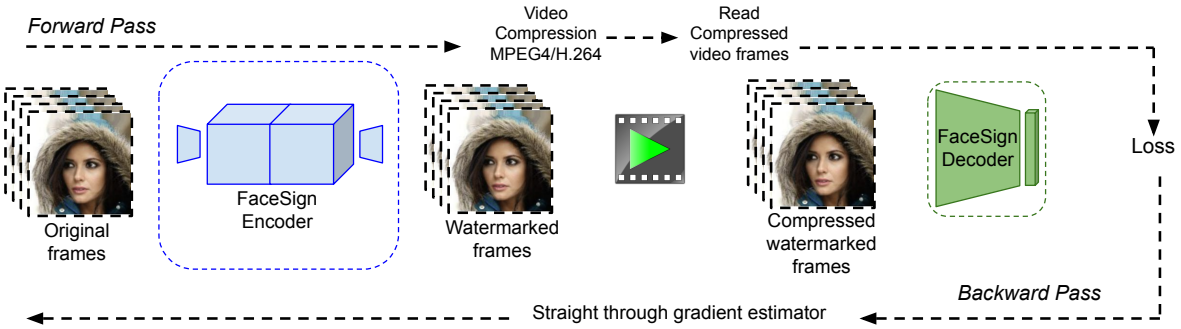


Figure 4.3. Training procedure to make watermarks robust against video compression codecs. We use the actual implementation of the video compression codec in the forward pass, and estimate the gradients in the backward pass using a straigh-through estimator.

We illustrate the video compression procedure used during training in Figure 4.3.

Malicious Transforms

The semi-fragile watermarks have to be unrecoverable when malicious transforms such as image compositing, occlusion or face replacement are applied. The common operation across these manipulation techniques is to modify certain spatial areas of the image. To simulate such transforms during training, we propose a *watermark occlusion transform* as follows: We first generate a tampering mask which indicates what modifications we want to retain or partially discard in the signed image. Given such a tampering mask, we partially remove the added perturbation in the signed image from the areas indicated by the mask. We consider two kinds of spatial tampering masks during training:

- **Image Compositing mask:** For each image, we initialize a mask $M_{h\times w\times c}$ of all ones. Next, we randomly select n rectangular patches in the mask and set the value of all pixels in the patches to a small watermark retention percentage $w_r \in [0,1]$.

- **Facial manipulation mask:** For each image, we initialize a mask $M_{h\times w\times c}$ of all ones. Next, we extract the facial feature polygons for *eyes*, *nose* and *lips* and set the values for all pixels inside the polygons to a small watermark retention percentage $w_r \in [0,1]$.

That is, That is, $M[i, j, :] = w_r$ for all i, j in the selected spatial polygons. Finally, the maliciously transformed image $g_m(x_w)$ is obtained as follows:

$$g_m(x_w) = M \cdot x_w + (1 - M) \cdot x$$

Figure 4.4 illustrates the malicious transform procedure.

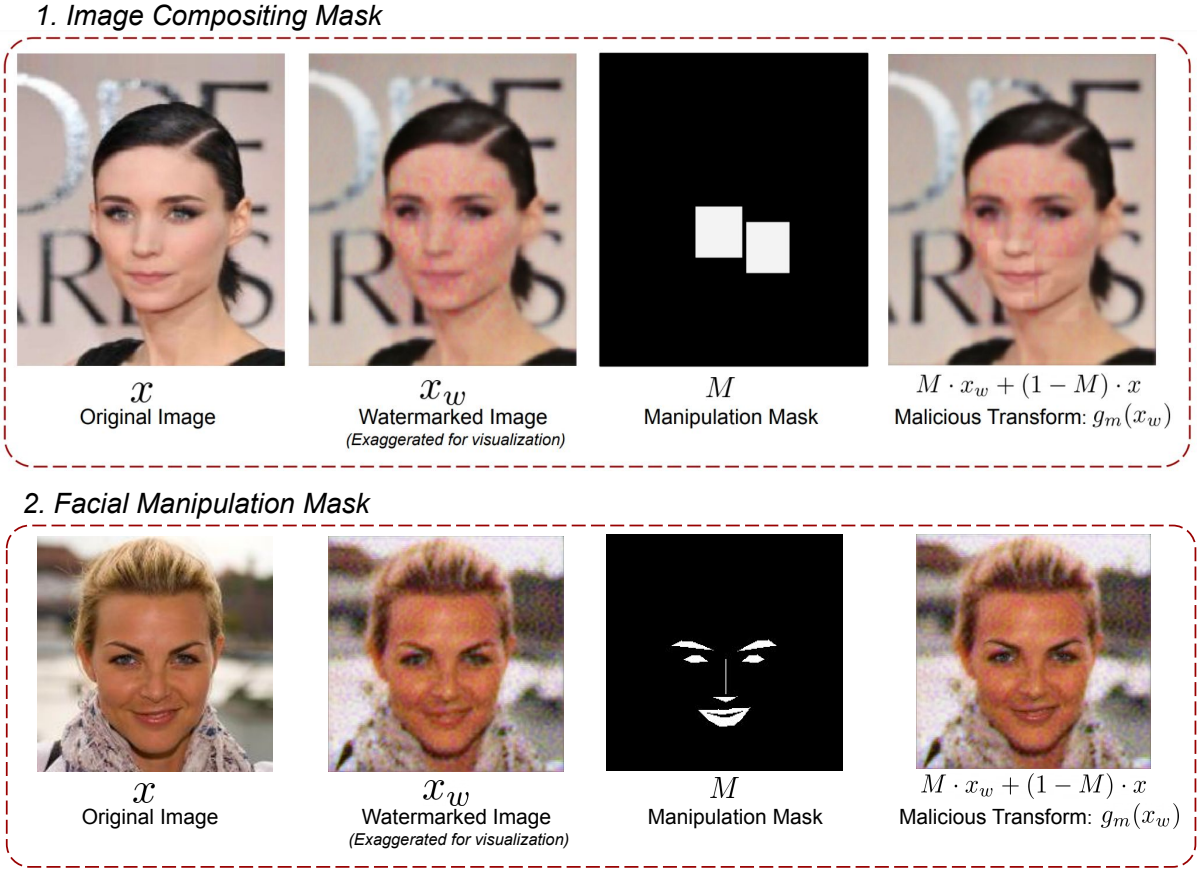


Figure 4.4. Malicious Transform: To simulate image tampering during training, the watermark is partially removed from the areas indicated by a manipulation mask.

4.3 Experiments

4.3.1 Datasets and Experimental Setup

We conduct our experiments on the CelebA dataset [105], MIRFLICKR [69] and UCF-101 [150] dataset. CelebA is a large scale database of over $200,000$ face images of $10,000$ unique celebrities. MIRFLICKR dataset is a diverse image retrieval dataset containing 1 million images. For training our watermarking framework to be robust to video compression, we use the UCF-101 dataset that contains 13320 short clips for action recognition. We set aside 1000 images/videos for testing from each dataset and split the remaining data into 80% training and 20% validation. We train our models for $200K$ mini-batch iterations with a batch-size of 64 and use an Adam optimizer with a fixed learning rate of $2e-4$. All our models are trained using images/video frames of size 256×256 which are obtained after center-cropping and resizing the images. We conduct experiments with message length $L = 128$. To evaluate the effectiveness of using transformation functions during training, we conduct an ablation study by training a *FaceSigns (No Transform)* model that does not incorporate any input transformations and a *FaceSigns (Robust)* model that uses only benign transformations during training. We evaluate watermarking techniques primarily on the following aspects:

1. **Imperceptibility:** We compare the original and watermarked images to compute: **peak signal to noise ratio (PSNR)** and **structural similarity index (SSIM)**. Higher values for both PSNR and SSIM are desirable for a more imperceptible watermark.

2. **Robustness and Fragility:** To measure the robustness and fragility of the watermarking system we measure the **bit recovery accuracy (BRA)** of the bit string s when unseen (not used in training) benign and malicious image transformations are applied. BRA is calculated by comparing the decoded secret bit-string with the secret bit-string that was embedded by the encoder into the given image. The number of matched bits divided by the length of the bit-string gives the bit recovery accuracy of a single image. We average

this over our test set to report the BRA. For robustness, it is desirable to have a high BRA against benign transformations like social media filters and image compression. For fragility against malicious tampering, it is desirable to have a low BRA when facial manipulation or image compositing is applied. To make a fair comparison with past works, we do not apply any bit error correcting codes while calculating the BRA and compare the input string s with the raw decoder output. A detector can classify an input as manipulated if the BRA of the decoded message is below a set threshold and benign if the BRA is more than the threshold. We measure the performance of such a detector using the **AUC score - Area under the ROC curve**.

3. **Capacity:** measures the amount of information that can be embedded in the image. We measure the capacity as the **bits per pixel (BPP)** that is the number of bits of the encrypted message embedded per pixel of the image which is simply $= L/(HWC)$.

It is important to note the trade-off between the above metrics—e.g. models with higher capacity, sacrifice on the imperceptibility or bit recovery accuracy. Similarly, more robust models sacrifice capacity or imperceptibility. We compare FaceSigns against three prior works on image watermarking — a DCT-based semi-fragile watermarking system [67] and two neural image watermarking systems HiDDeN [191] and StegaStamp [160]. Both HiDDeN and StegaStamp embed a bit string message into a square RGB image while ensuring robustness to a set of image transformations. We present examples of original and watermarked images along with the added perturbation from different techniques in Figure 4.5.

4.3.2 Imperceptibility and Capacity

The image similarity and capacity metrics of different watermarking techniques are reported in Table 4.1. We find that even at a higher message capacity, FaceSigns can encode messages with better imperceptibility as compared to StegaStamp and HiDDeN. As noted by the authors of StegaStamp and visible in Figure 4.5 and Figure 4.10 , the residual added by

Figure 4.5. Examples of original and watermarked images using prior works and our *FaceSigns (Semi-Fragile)* model. The image perturbation has been linearly scaled between 0 and 1 for visualization.

their model is perceptible in large low frequency regions of the image. We believe that this is primarily due to the difference in our network architecture choices. In our initial experiments, we found that using a UNet architecture for the decoder with an intermediate message reconstruction loss described in Section 4.2.2, performed significantly better than a down-sampling CNN architecture used in prior work. Additionally, we use nearest neighbour upsampling instead of transposed convolutions in our U-Net architectures which helps reduce the perceptibility of the watermark by removing upsampling artifacts.

4.3.3 Robustness and Fragility

To study the robustness and fragility of different DNN based watermarking techniques, we transform the watermarked images using unseen benign and malicious transformations and then attempt to decode the message from the transformed message. We perform ablation studies to evaluate the effectiveness of the proposed transforms by training three versions of our watermarking framework: *FaceSigns (No Transform)* that does not use any benign or malicious transformations during training, *FaceSigns (Robust)* that is only trained to be robust against

Table 4.1. Capacity and imperceptibility metrics of different watermarking systems. H, W indicate the height and width of the input image.

	Capacity			*Imperceptibility*	
Method	H,W	L	BPP	PSNR	SSIM
SemiFragile DCT [67]	128	256	5.2e-3	22.49	0.871
HiDDeN [191]	128	30	6.1e-4	27.57	0.934
StegaStamp [160]	400	100	2.0e-4	29.39	0.925
FaceSigns (No Transform)	256	128	6.5e-4	36.38	0.973
FaceSigns (Robust)	256	128	6.5e-4	35.56	0.964
FaceSigns (Semi-Fragile)	256	128	6.5e-4	35.43	0.962

benign transformations and does not use malicious transformations during training and *FaceSigns (Semi-Fragile)* which uses both benign and malicious transformations during training.

Benign Image Transforms:

For benign transforms, we consider real-world image operations that are commonly used when uploading pictures on the internet. We compress the image using different levels of JPEG compression (separate from training) and also apply Instagram filters namely *Aden*, *Brooklyn* and *Clarendon* which we use from an open-source python library - Pilgram [78]. Some example images from these transformations are shown in Figure 4.6. We report the BRA of different watermarking frameworks after undergoing benign transformations in Table 4.2. We find that both StegaStamp and our robust and semi-fragile models can decode secrets with a high BRA for these image transformations. We find that *FaceSigns (Robust)*, which does not use malicious transforms during training, is slightly more robust to benign transformations as compared to *FaceSigns (Semi-Fragile)*. However, this improved robustness comes at the cost of being non-fragile to malicious transformations and being able to decode messages with high BRA even for Deepfake manipulations. The model *FaceSigns (No-Transform)* which does not incorporate any benign or malicious transformations during training is fragile to both JPEG compression and malicious transforms as indicated by the low BRA for both methods.

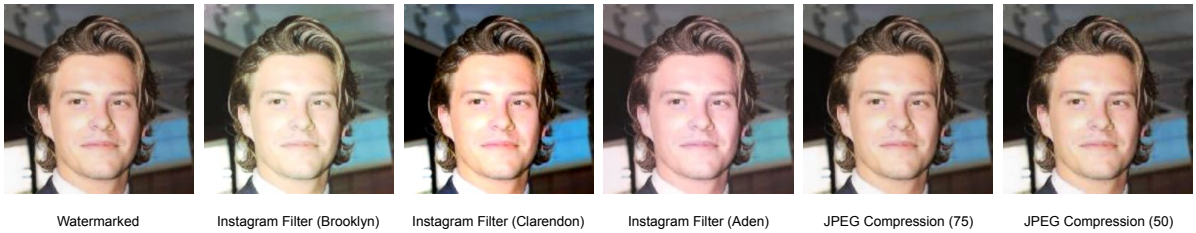


Figure 4.6. Watermarked images with unseen benign transformations applied. Benign transformations depicted in this diagram include Instagram filters [78] Brooklyn, Clarendon, Aden and various levels of JPEG compression

Robustness to Video Compression: For watermarking videos, we use the FaceSigns encoder to insert the watermark data into each video frame. Similarly for decoding, we decode watermark data by passing each frame of the watermarked video to the FaceSigns decoder network. In our initial experiments, we found that training FaceSigns to be robust against spatial image transforms does not ensure robustness against video compression codecs. This is because besides compressing each frame spatially, video compression codecs like H264 also compress data temporally. To address this challenge, we incorporate video compression during training using the gradient-estimation procedure described in Section 4.2.3. As indicated by the results in Figure 4.7-C, incorporating video compression codecs during training significantly improves watermark recovery from highly compressed videos. Robustness to H264 compression makes FaceSigns a practical framework for inserting recoverable watermarks in videos shared on the internet.

Table 4.2. Bit recovery accuracy (BRA) of different watermarking techniques against benign and malicious transforms. A higher BRA against benign and a lower BRA against malicious transforms is desirable to achieve our goal of semi-fragile watermarking.

	BRA (%) - Benign Transforms						*BRA (%) - Malicious Transforms*			
Method	None	JPG-75	JPG-50	*Aden*	*Brooklyn*	*Clarendon*	SimSwap [26]	FSFT [144]	FS [89]	Compositing
SemiFragile DCT [67]	99.81	56.65	55.04	94.98	96.41	95.06	57.62	57.61	88.59	82.31
HiDDeN [191]	97.06	72.71	68.48	94.52	94.52	94.52	85.48	72.33	74.23	73.27
StegaStamp [160]	99.92	99.91	99.87	99.84	99.73	99.39	98.34	97.42	97.43	98.21
FaceSigns (No Transform)	99.96	50.51	50.07	98.39	99.67	99.65	51.04	52.00	51.36	53.36
FaceSigns (Robust)	99.96	99.74	97.26	99.53	99.19	99.37	97.29	89.76	68.99	97.26
FaceSigns (Semi-Fragile)	99.68	99.49	98.38	97.40	98.34	99.32	64.93	52.21	31.77	51.61

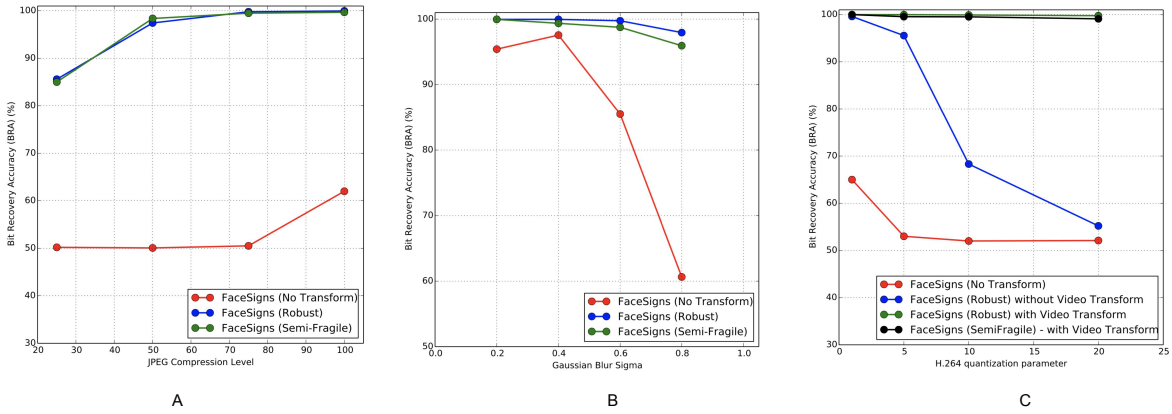


Figure 4.7. Fig. A: BRA vs JPEG compression levels (lower values indicate higher compression). Fig. B: BRA vs sigma value used for Gaussian blur (higher sigma corresponds to higher distortion). Fig. C. BRA vs quantization factor for H264 video codec.

Malicious Transforms:

To evaluate fragility of the watermark against unseen facial manipulations, we apply three face-swapping techniques on the CelebA watermarked images: FaceSwap [89], SimSwap [26] and Few-Shot Face Translation (FSFT) [144]. FaceSwap [89] is a computer graphics based technique that swaps the face by aligning the facial landmarks of the two images. SimSwap [26] and FSFT [144] are deep learning based techniques that use CNN encoder-decoder networks trained using adversarial loss to generate Deepfakes. Figure 4.9 shows examples of swapped faces using these techniques. Additionally, we consider a general image compositing operation for all test images where we randomly select image patches covering 10% to 50% of the image and replace the patches with those from an alternate image.

As reported in Table 4.2, we find that StegaStamp and *FaceSigns (Robust)* can decode signatures from maliciously transformed images with a high BRA thereby making them unsuitable for authenticating the integrity of digital media. This is understandable since these methods prioritize robustness over fragility. StegaStamp has been shown to be robust to occlusions even though occlusions were not explicitly a part of their set of training transformations. In contrast, watermark data recovery for *FaceSigns (Semi-Fragile)* model breaks against malicious transforms, which is desirable for malicious tampering detection.

Based on the bit-recovery accuracy of the watermark data, we can define a manipulation detector as follows: The detector labels an image as maliciously tampered if the BRA of the predicted bit-string is less than a threshold τ. The ROC curve of such a detector is shown in Figure 4.8. As evident by the ROC plots and AUC scores shown in Figure 4.8, in contrast to prior works, our semi-fragile model demonstrates robustness to benign transformations while being fragile toward out-of-domain malicious Deepfake transformations, thereby achieving our goal of selective fragility and an AUC score of 0.996 for manipulation detection.

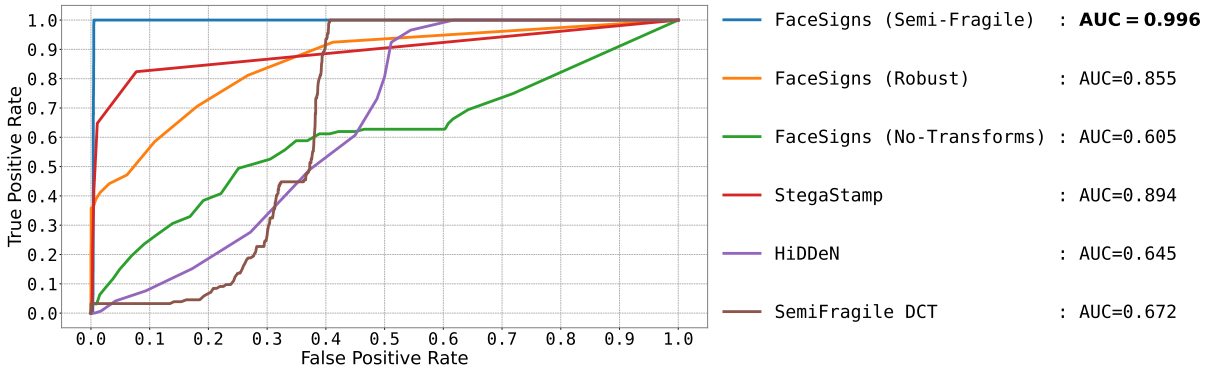


Figure 4.8. Manipulation detection ROC plots and AUC scores for different watermarking techniques. The watermarking framework labels an example as manipulated if the BRA for an image is less than a given threshold.

Figure 4.9. Facially manipulated images created through SimSwap [26], FSFT [144] and FaceSwap [89] techniques for evaluating the fragility of the watermark.

4.4 Discussion - Threat Models

Media authentication systems face adversarial threats from attackers who attempt to bypass the detectors by authenticating manipulated media. In this section, I discuss some of the threat models faced by our system and how these challenges can be addressed:

Attack 1. **Querying the decoder network for performing adversarial attacks:** The attacker may query the decoder network with an image to get the decoded message and adversarially perturb the query image until the decoded message matches the target message.

Defense: The attacker does not know what target messages can prove media authenticity since these messages can be kept as a secret and updated frequently. If the attacker gains access to the secret message by querying the decoder with a watermarked image, the encryption key secrecy can prevent the attacker from knowing the target encrypted message for the decoder. Lastly, the decoder network can be hosted securely and can only output a binary label indicating whether the image is authentic or manipulated by matching the decoded secret with the list of trusted secrets. This would make the decoder's signal unusable for performing adversarial attacks to match a target message out of the total possible 2^{128} messages.

Attack 2. **Copying the watermark perturbation from one image to another:** The adversary may attempt to extract the added perturbation of the watermark and add it onto a Deepfake image to authenticate the manipulated media.

Defense: Since FaceSigns generates an image and message specific perturbation, we hypothesize that the same perturbation when applied on alternate images should not be recoverable by the decoder. We verify this hypothesis by conducting an experiment in which we extract added perturbations from 100 watermarked images, and apply extracted perturbation to 100 alternate images. The bit recovery accuracy of such an attack is just 17.6% which is worse than random prediction.

Figure 4.10. Additional examples of original and watermarked images using prior works and our method (FaceSigns). Observe the change in the perturbation pattern as we incorporate both robust and benin transformations in the *FaceSigns (Semi-Fragile)* model.

Attack 3. **Training a proxy encoder:** The adversary can collect a dataset of original and watermarked images and train a neural network based encoder-decoder image-to-image translation network to map any new image to a watermarked image.

Defense: One defense strategy is to only store watermarked images on devices so that an attacker never gains access to pairs of original and watermarked images. Also, the above attack can only work if the encoded images all contain the same secret message, so that the adversary can learn a generator for watermarking a new image with the same secret message. To prevent the creation of such a dataset, some bits of the message can be kept dynamic and contain a unique time-stamp and device-specific codes so that each embedded bit-string is different. Regularly updating the trusted message or encryption key is another preventative strategy against such attacks.

4.5 Conclusion

In this chapter, we described FaceSigns, a deep learning based semi-fragile watermarking system that can certify the integrity of digital images and videos, and reliably detect tampering. Through our experiments and evaluations, we demonstrate that FaceSigns generates more imperceptible watermarks than previous state-of-the-art methods while upholding the desired semi-fragile characteristics. By carefully designing a fixed set of benign and malicious transformations during training, our framework achieves generalizability to real-world image and video transformations and can reliably detect Deepfake facial and image compositing manipulations unlike prior image watermarking techniques. Additionally, our work is a significant step forward in the field of covert watermarking for videos. FaceSigns can be vital to media authenticators in social media platforms, news agencies and legal offices and help create more trustworthy platforms and establish consumer trust in digital media.

4.6 Acknowledgements

Chapter 4 is a reprint of the material as it appears in *FaceSigns: Semi-Fragile Neural Watermarks for Media Authentication and Countering Deepfakes*. 2023. Neekhara, Paarth; Hussain, Shehzeen; Zhang, Xinqiao; Koushanfar, Farinaz; McAuley, Julian. Currently under review for publication. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Conclusion

In this dissertation, I discussed my research on developing expressive speech synthesis frameworks and discussed methodologies for detecting synthesized media. On the synthesis side, I developed two frameworks that allow the user to synthesize a given voice from either text (voice cloning) or a reference speech input (voice conversion). Both these systems generate expressive and natural sounding speech that is suitable for accompanying AI generated visual content. Next, I discussed the potential misuse and harms of high-quality media synthesis technology and the need to develop reliable detection methods. My research explored the limitations in existing Deepfake detectors and exposed major vulnerabilities using adversarial examples. Finally, to address these limitations, I described a semi-fragile image and video watermarking framework I developed for authentication real images and videos.

Generation and Detection of synthesized media are both important problems that need to be addressed together for the responsible use of media-synthesis technology. As AI generated content is expected to increase across social media platforms in the foreseeable future, reliable detection of such content is essential to ensure trust in social media platforms and prevent potential harms of the synthesis technology. I plan to extend my research and develop media authentication and detection systems for audio and speech domain.

Bibliography

[1] 100,000 Faces Generated by AI, 2018.

[2] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. MesoNet: a Compact Facial Video Forgery Detection Network. In *Proc. IEEE International Workshop on Information Forensics and Security*, 2018.

[3] Sercan Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. Neural voice cloning with a few samples. In *NeurIPS*. 2018.

[4] Sercan Ö Arik, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Andrew Ng, and Jonathan Raiman. Deep Voice: Real-time neural text-to-speech. In Proc. *ICML*, pages 195–204. JMLR. org, 2017.

[5] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

[6] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *NeurIPS*, 2019.

[7] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *NeurIPS*, 2020.

[8] Shumeet Baluja. Hiding images in plain sight: Deep steganography. *NIPS*, 2017.

[9] Jawadul H Bappy, Amit K Roy-Chowdhury, Jason Bunk, Lakshmanan Nataraj, and BS Manjunath. Exploiting spatial structure for localizing manipulated image regions. In *Proceedings of the IEEE international conference on computer vision*, pages 4970–4979, 2017.

[10] Jawadul H Bappy, Cody Simons, Lakshmanan Nataraj, BS Manjunath, and Amit K Roy-Chowdhury. Hybrid lstm and encoder–decoder architecture for detection of image forgeries. *IEEE Transactions on Image Processing*, 28(7):3286–3300, 2019.

[11] Mauro Barni, Matthew C Stamm, and Benedetta Tondi. Adversarial multimedia forensics: Overview and challenges ahead. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pages 962–966. IEEE, 2018.

[12] Melika Behjati, Seyed-Mohsen Moosavi-Dezfooli, Mahdieh Soleymani Baghshah, and Pascal Frossard. Universal adversarial attacks on text classifiers. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7345–7349, 2019.

[13] Yi-Lin Bei, Xiao-Rong Zhu, Qian Zhang, and Sai Qiao. A robust image watermarking algorithm based on content authentication and intelligent optimization. In *Proceedings of the 5th International Conference on Control and Computer Vision*, ICCCV '22, page 164–170, New York, NY, USA, 2022. Association for Computing Machinery.

[14] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[15] Oussama Benrhouma, Houcemeddine Hermassi, and Safya Belghith. Tamper detection and self-recovery scheme by dwt watermarking. *Nonlinear Dynamics*, 2015.

[16] Siddharth Bhalerao, Irshad Ahmad Ansari, and Anil Kumar. A secure image watermarking for tamper detection and localization. *Journal of Ambient Intelligence and Humanized Computing*, 2021.

[17] Ning Bi, Qiyu Sun, Daren Huang, Zhihua Yang, and Jiwu Huang. Robust image watermarking based on multiband wavelets and empirical mode decomposition. *IEEE Transactions on Image Processing*, 2007.

[18] R Bohme and M Kirchner. Digital image forensics: There is more to a picture than meets the eye, chapter counter-forensics: Attacking image forensics, 2013.

[19] Rainer Böhme and Matthias Kirchner. Counter-forensics: Attacking image forensics. In *Digital image forensics*, pages 327–366. Springer, 2013.

[20] Q. Cao, , L. Shen, W. Xie, O.M. Parkhi, and A. Zisserman. VGGFace2: A Dataset for Recognising Faces Across Pose and Age. In *Proc. International Conference on Automatic Face & Gesture Recognition*, 2018.

[21] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (sp)*, pages 39–57. IEEE, 2017.

[22] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020.

[23] E. Casanova, J. Weber, C.D. Shulby, A.C. Junior, E. Gölge, and M. A. Ponti. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In *ICML*. PMLR, 2022.

[24] CelebA-HQ, 2018.

[25] Ling-Hui Chen, Zhen-Hua Ling, Li-Juan Liu, and Li-Rong Dai. Voice conversion using deep neural networks with layer-wise generative training. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(12):1859–1872, 2014.

[26] Renwang Chen, Xuanhong Chen, Bingbing Ni, and Yanhao Ge. Simswap: An efficient framework for high fidelity face swapping. In *MM '20: The 28th ACM International Conference on Multimedia*. ACM, 2020.

[27] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*. PMLR, 2020.

[28] H. S. Choi, J. Lee, W. Kim, J. Lee, H. Heo, and K. Lee. Neural analysis and synthesis: Reconstructing speech from self-supervised representations. *NeurIPS*, 2021.

[29] Hyeong-Seok Choi, Jinhyeok Yang, Juheon Lee, and Hyeongju Kim. NANSY++: Unified voice synthesis with neural analysis and synthesis. In *ICLR*, 2023.

[30] Y. Choi, M. Choi, M. Kim, J.W. Ha, S. Kim, and J. Choo. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[31] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[32] F. Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[33] J. Chou and H. Y. Lee. One-shot voice conversion by separating speaker and content representations with instance normalization. *Interspeech*, 2019.

[34] Wei Chu and Abeer Alwan. Reducing f0 frame error of f0 tracking algorithms under noisy conditions with an unvoiced/voiced classification frontend. In *ICASSP*. IEEE, 2009.

[35] J. S. Chung, A. Nagrani, and A. Zisserman. Voxceleb2: Deep speaker recognition. In *INTERSPEECH*, 2018.

[36] Ingemar J Cox, Joe Kilian, F Thomson Leighton, and Talal Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE transactions on image processing*, 1997.

[37] Ingemar J Cox, Matthew L Miller, Jeffrey Adam Bloom, and Chris Honsinger. *Digital watermarking*, volume 53. Springer, 2002.

[38] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. Jain. On the Detection of Digital Face Manipulation. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[39] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E Kounavis, and Duen Horng Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*, 2017.

[40] Azat Davletshin. https://github.com/ntech-lab/deepfake-detection-challenge, 2020.

[41] A. De Cheveigné and H. Kawahara. Yin, a fundamental frequency estimator for speech and music. In *The Journal of the Acoustical Society of America*, 2002.

[42] DeepFakes. https://github.com/deepfakes/faceswap, 2017.

[43] Jiankang Deng, Jia Guo, Evangelos Ververas, Irene Kotsia, and Stefanos Zafeiriou. RetinaFace: Single-shot multi-level face localisation in the wild. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[44] Ferdinando Di Martino and Salvatore Sessa. Fragile watermarking tamper detection via bilinear fuzzy relation equations. *Journal of Ambient Intelligence and Humanized Computing*, 2019.

[45] Brian Dolhansky, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. The DeepFake Detection Challenge (DFDC) dataset. *arXiv preprint arXiv:2006.07397*, 2020.

[46] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) dataset. *arXiv preprint arXiv:2006.07397*, 2020.

[47] Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *ICLR*, 2019.

[48] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018.

[49] Yinpeng Dong, Tianyu Pang, Hang Su, and Jun Zhu. Evading defenses to transferable adversarial examples by translation-invariant attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[50] Amanda Duarte, Francisco Roldan, Miquel Tubau, Janna Escur, Santiago Pascual, Amaia Salvador, Eva Mohedano, Kevin McGuinness, Jordi Torres, and Xavier Giro-i Nieto. Wav2pix: speech-conditioned face generation using generative adversarial networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 3, 2019.

[51] Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M Roy. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016.

[52] Hany Farid. *Photo Forensics*. The MIT Press, 2016.

[53] Jessica Fridrich. *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009.

[54] Andrew Gibiansky, Sercan Arik, Gregory Diamos, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. Deep voice 2: Multi-speaker neural text-to-speech. In *NIPS*, 2017.

[55] E. Gonzalez-Sosa, J. Fierrez, R. Vera-Rodriguez, and F. Alonso-Fernandez. Facial Soft Biometrics for Recognition in the Wild: Recent Works, Annotation and COTS Evaluation. *IEEE Transactions on Information Forensics and Security*, 13(8):2001–2014, 2018.

[56] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In Proc. *NIPS*, pages 2672–2680, 2014.

[57] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *stat*, 2015.

[58] Daniel W. Griffin, Jae, S. Lim, and Senior Member. Signal estimation from modified short-time Fourier transform. *IEEE Trans. Acoustics, Speech and Sig. Proc*, 1984.

[59] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, and Mohammad Gheshlaghi Azar. Bootstrap your own latent-a new approach to self-supervised learning. *NeurIPS*, 2020.

[60] Y. Gu, Z. Zhang, X. Yi, and X. Zhao. Mediumvc: Any-to-any voice conversion using synthetic specific-speaker speeches as intermedium features. *arXiv:2110.02500*, 2021.

[61] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[62] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, and Y. Wu. Conformer: Convolution-augmented transformer for speech recognition. *Interspeech*, 2020.

[63] Bin Guo, Yasan Ding, Lina Yao, Yunji Liang, and Zhiwen Yu. The future of false information detection on social media: New perspectives and trends. *ACM Comput. Surv.*, 53(4), 2020.

[64] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[65] Cui Hao. https://github.com/cuihaoleo/kaggle-dfdc, 2020.

[66] Jamie Hayes and George Danezis. Generating steganographic images via adversarial training. In *International Conference on Neural Information Processing Systems*, 2017.

[67] Chi Kin Ho and Chang-Tsun Li. Semi-fragile watermarking scheme for authentication of jpeg images. In *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.* IEEE, 2004.

[68] W. C. Huang, S. W. Yang, T. Hayashi, H. Y. Lee, S. Watanabe, and T. Toda. S3prl-vc: Open-source voice conversion framework with self-supervised speech representations. In *ICASSP*. IEEE, 2022.

[69] Mark J. Huiskes and Michael S. Lew. The mir flickr retrieval evaluation. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, MIR '08, New York, NY, USA, 2008. Association for Computing Machinery.

[70] S. Hussain, V. Nguyen, S. Zhang, and E. Visser. Multi-task voice activated framework using self-supervised learning. In *ICASSP*. IEEE, 2022.

[71] Shehzeen Hussain, Paarth Neekhara, Jocelyn Huang, Jason Li, and Boris Ginsburg. Ace-vc: Adaptive and controllable voice conversion using explicitly disentangled self-supervised speech representations. In *ICASSP*. IEEE, 2023.

[72] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146, 2018.

[73] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[74] Keith Ito. The lj speech dataset. https://keithito.com/LJ-Speech-Dataset/, 2017.

[75] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, zhifeng Chen, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, and Yonghui Wu. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. In *NeurIPS*. 2018.

[76] Z. Jin, J. Cao, Han Guo, Yongdong Zhang, Y. Wang, and Jiebo Luo. Detection and analysis of 2016 us presidential election related rumors on twitter. In *SBP-BRiMS*, 2017.

[77] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P. E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP*, 2020.

[78] Akiomi Kamakura. pilgram https://github.com/akiomik/pilgram.

[79] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *Proc. International Conference on Learning Representations*, 2018.

[80] T. Karras, S. Laine, and T. Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[81] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[82] Simon J. King and Vasilis Karaiskos. The blizzard challenge 2013. In *In Blizzard Challenge Workshop*, 2013.

[83] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[84] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[85] Nithin Rao Koluguri, Taejin Park, and Boris Ginsburg. Titanet: Neural model for speaker representation with 1d depth-wise separable convolutions and global context. In *ICASSP*, 2022.

[86] N.R. Koluguri, J. Li, V. Lavrukhin, and B. Ginsburg. Speakernet: 1d depth-wise separable convolutional network for text-independent speaker recognition and verification. *arXiv:2010.12653*, 2020.

[87] J. Kong, J. Kim, and J. Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *NeurIPS*, 2020.

[88] P. Korshunov and S. Marcel. Deepfakes: a New Threat to Face Recognition? Assessment and Detection. *arXiv preprint arXiv:1812.08685*, 2018.

[89] Marek Kowalski. Faceswap https://github.com/marekkowalski/faceswap/, 2018.

[90] S. Kriman, S. Beliaev, B. Ginsburg, J. Huang, O. Kuchaiev, V. Lavrukhin, R. Leary, J. Li, and Y. Zhang. Quartznet: Deep automatic speech recognition with 1d time-channel separable convolutions. In *ICASSP*. IEEE, 2020.

[91] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[92] K. Lakhotia, E. Kharitonov, W.N. Hsu, Y. Adi, A. Polyak, B. Bolte, T. Nguyen, J. Copet, A. Baevski, and A. Mohamed. On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 2021.

[93] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. Denoyer, and M.A. Ranzato. Fader Networks: Manipulating Images by Sliding Attributes. In *Proc. Advances in Neural Information Processing Systems*, 2017.

[94] A. Łańcucki. Fastpitch: Parallel text-to-speech with pitch prediction. In *ICASSP*. IEEE, 2021.

[95] Jonathan Le Roux, Hirokazu Kameoka, Nobutaka Ono, and Shigeki Sagayama. Fast signal reconstruction from magnitude STFT spectrogram based on spectrogram consistency. In Proc. *International Conference on Digital Audio Effects*, pages 397–403, 2010.

[96] Chunlei Li, Aihua Zhang, Zhoufeng Liu, Liang Liao, and Di Huang. Semi-fragile self-recoverable watermarking algorithm based on wavelet group quantization and double authentication. *Multimedia tools and applications*, 2015.

[97] Jian Li, Yabiao Wang, Changan Wang, Ying Tai, Jianjun Qian, Jian Yang, Chengjie Wang, Ji-Lin Li, and Feiyue Huang. DSFD: dual shot face detector. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[98] Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, and Baining Guo. Face x-ray for more general face forgery detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5001–5010, 2020.

[99] Y. Li, M.C. Chang, and S. Lyu. In Ictu Oculi: Exposing AI Generated Fake Face Videos by Detecting Eye Blinking. In *Proc. IEEE International Workshop on Information Forensics and Security*, 2018.

[100] Eugene T Lin, Christine I Podilchuk, and Edward J Delp III. Detection of image alterations using semifragile watermarks. In *Security and Watermarking of Multimedia Contents II*. International Society for Optics and Photonics, 2000.

[101] Y. Lin, C. M. Chien, J. H. Lin, H. Lee, and L. S. Lee. Fragmentvc: Any-to-any voice conversion by end-to-end extracting and fusing fine-grained voice fragments with attention. In *ICASSP*. IEEE, 2021.

[102] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*. IEEE, 2017.

[103] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[104] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep Learning Face Attributes in the Wild. In *Proc. IEEE/CVF International Conference on Computer Vision*, 2015.

[105] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, December 2015.

[106] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[107] Gilles Louppe. Master thesis : Automatic multispeaker voice cloning. 2019.

[108] Gilles Louppe. *Resemblyzer - https://github.com/resemble-ai/Resemblyzer/*, 2019.

[109] Xiyang Luo, Ruohan Zhan, Huiwen Chang, Feng Yang, and Peyman Milanfar. Distortion agnostic deep watermarking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[110] Matthias Mauch and Simon Dixon. Pyin: A fundamental frequency estimator using probabilistic threshold distributions. In *ICASSP*, 2014.

[111] Yisroel Mirsky and Wenke Lee. The creation and detection of deepfakes: A survey. *ACM Comput. Surv.*, 54(1), jan 2021.

[112] M. Mirza and Simon S. Osindero. Conditional Generative Adversarial Nets. *arXiv preprint arXiv:1411.1784*, 2014.

[113] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *CVPR*, 2020.

[114] S. H. Mohammadi and A. Kain. An overview of voice conversion systems. In *Speech Communication*. Elsevier, 2017.

[115] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[116] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman. Voxceleb: Large-scale speaker verification in the wild. *Computer Science and Language*, 2019.

[117] Tomohiro Nakatani, Shigeaki Amano, Toshio Irino, Kentaro Ishizuka, and Tadahisa Kondo. A method for fundamental frequency estimation and voicing decision: Application to infant utterances recorded in real acoustical environments. *Speech Communication*, 2008.

[118] Paarth Neekhara, Chris Donahue, Miller Puckette, Shlomo Dubnov, and Julian McAuley. Expediting TTS synthesis with adversarial vocoding. In *INTERSPEECH*, 2019.

[119] Paarth Neekhara, Shehzeen Hussain, Prakhar Pandey, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. Universal adversarial perturbations for speech recognition systems. In *Interspeech*, 2019.

[120] CBS News. *Doctored Nancy Pelosi video highlights threat of "deepfake" tech*, 2019.

[121] Yuval Nirkin, Yosi Keller, and Tal Hassner. FSGAN: Subject agnostic face swapping and reenactment. In *CVPR*, 2019.

[122] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[123] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *ICASSP*. IEEE, 2015.

[124] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 2017.

[125] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[126] Kyubyong Park and Thomas Mulc. Css10: A collection of single speaker speech datasets for 10 languages. *Interspeech*, 2019.

[127] G. Perarnau, J. Van De Weijer, B. Raducanu, and J.M. Álvarez. Invertible Conditional GANs for Image Editing. In *Proc. Advances in Neural Information Processing Systems Workshops*, 2016.

[128] Shelby Pereira and Thierry Pun. Robust template matching for affine resistant image watermarks. *IEEE transactions on image Processing*, 2000.

[129] Shelby Pereira, Joseph JK O Ruanaidh, Frederic Deguillaume, Gabriela Csurka, and Thierry Pun. Template based recovery of fourier-based watermarks using log-polar and log-log maps. In *Proceedings IEEE International Conference on Multimedia Computing and Systems*. IEEE, 1999.

[130] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. Deep Voice 3: Scaling text-to-speech with convolutional sequence learning. In *ICLR*, 2018.

[131] A. Polyak, Y. Adi, J. Copet, E. Kharitonov, K. Lakhotia, W.N. Hsu, A. Mohamed, and E. Dupoux. Speech resynthesis from discrete disentangled self-supervised representations. In *Interspeech*, 2021.

[132] RO Preda and DN Vizireanu. Watermarking-based image authentication robust to jpeg compression. *Electronics Letters*, 2015.

[133] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. WaveGlow: A flow-based generative network for speech synthesis. In *ICASSP*, 2018.

[134] K. Qian, Y. Zhang, S. Chang, X. Yang, and M. Hasegawa-Johnson. Autovc: Zero-shot voice style transfer with only autoencoder loss. In *ICML*. PMLR, 2019.

[135] K. Qian, Y. Zhang, H. Gao, J. Ni, C.I. Lai, D. Cox, M. Hasegawa-Johnson, and S. Chang. Contentvec: An improved self-supervised speech representation by disentangling speakers. In *International Conference on Machine Learning*. PMLR, 2022.

[136] Ramachandra Raghavendra, Kiran B Raja, Sushma Venkatesh, and Christoph Busch. Transferable deep-cnn features for detecting digital and print-scanned morphed face images. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1822–1830. IEEE, 2017.

[137] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[138] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

[139] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Niessner. Faceforensics++: Learning to detect manipulated facial images. In *ICCV*, October 2019.

[140] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, and Prem Natarajan. Recurrent convolutional strategies for face manipulation detection in videos. *Interfaces (GUI)*, 3:1, 2019.

[141] D. Saito, K. Yamamoto, N. Minematsu, and K. Hirose. One-to-many voice conversion based on tensor representation of speaker space. In *Interspeech*, 2011.

[142] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution Strategies as a Scalable Alternative to Reinforcement Learning, September 2017.

[143] Selim Seferbekov. https://github.com/selimsef/dfdc_deepfake-_challenge, 2020.

[144] Shaoanlu. Few-shot face translation https://github.com/shaoanlu/fewshot-face-translation-gan, 2019.

[145] S Shefali and SM Deshpande. Information security through semi-fragile watermarking. In *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*. IEEE, 2007.

[146] Abdulaziz Shehab, Mohamed Elhoseny, Khan Muhammad, Arun Kumar Sangaiah, Po Yang, Haojun Huang, and Guolin Hou. Secure and robust fragile watermarking scheme for medical images. *IEEE Access*, 2018.

[147] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, and Rj Skerrv-Ryan. Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions. In *ICASSP*, 2018.

[148] Richard Shin and Dawn Song. Jpeg-resistant adversarial images. In *NIPS 2017 Workshop on Machine Learning and Computer Security*, 2017.

[149] R. J. Skerry-Ryan, Eric Battenberg, Ying Xiao, Yuxuan Wang, Daisy Stanton, Joel Shor, Ron J. Weiss, Rob Clark, and Rif A. Saurous. Towards end-to-end prosody transfer for expressive speech synthesis with tacotron. *arXiv:1803.09047*, 2018.

[150] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[151] Daisy Stanton, Yuxuan Wang, and R. J. Skerry-Ryan. Predicting expressive speaking style from text in end-to-end speech synthesis. *arXiv:1803.09017*, 2018.

[152] Stanley Smith Stevens, John Volkmann, and Edwin B Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

[153] L. Sun, K. Li, H. Wang, S. Kang, and H. Meng. Phonetic posteriorgrams for many-to-one voice conversion without parallel data training. In *ICME*, 2016.

[154] Lifa Sun, Shiyin Kang, Kun Li, and Helen Meng. Voice conversion using deep bidirectional long short-term memory based recurrent neural networks. In *ICASSP*, 2015.

[155] Rui Sun, Hong Sun, and Tianren Yao. A svd-and quantization based semi-fragile watermarking technique for image authentication. In *6th International Conference on Signal Processing, 2002.* IEEE, 2002.

[156] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Synthesizing obama: learning lip sync from audio. *ACM Transactions on Graphics (TOG)*, 2017.

[157] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[158] Alex Tamkin, Mike Wu, and Noah Goodman. Viewmaker networks: Learning views for unsupervised representation learning. In *ICLR*, 2021.

[159] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114, 2019.

[160] Matthew Tancik, Ben Mildenhall, and Ren Ng. Stegastamp: Invisible hyperlinks in physical photographs. In *CVPR*, 2020.

[161] J. Thies, M. Zollhöfer, and M. Nießner. Deferred Neural Rendering: Image Synthesis using Neural Textures. *ACM Transactions on Graphics*, 38(66):1–12, 2019.

[162] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *CVPR*, 2016.

[163] X. Tian, J. Wang, H. Xu, E. S. Chng, and H. Li. Average modeling approach to voice conversion with non-parallel data. In *Odyssey*, 2018.

[164] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *NeurIPS*, 2020.

[165] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, Aythami Morales, and Javier Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection, 2020.

[166] Cristian Vaccari and Andrew Chadwick. Deepfakes and disinformation: exploring the impact of synthetic political video on deception, uncertainty, and trust in news. *Social Media+ Society*, 2020.

[167] Rafael Valle, Jason Li, Ryan Prenger, and Bryan Catanzaro. Mellotron: Multispeaker expressive voice synthesis by conditioning on rhythm, pitch and global style tokens. *ICASSP*, 2020.

[168] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv:1609.03499*, 2016.

[169] Luisa Verdoliva. Media forensics and deepfakes: an overview. *arXiv preprint arXiv:2001.06564*, 2020.

[170] Konstantinos Vougioukas, Stavros Petridis, and Maja Pantic. Realistic speech-driven facial animation with gans. *International Journal of Computer Vision*, pages 1–16, 2019.

[171] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. Generalized end-to-end loss for speaker verification. *arXiv:1710.10467*, 2017.

[172] Weihong Wang and Hany Farid. Exposing digital forgeries in interlaced and deinterlaced video. *IEEE Transactions on Information Forensics and Security*, 2007.

[173] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, and Samy Bengio. Tacotron: Towards end-to-end speech synthesis. In Proc. *INTERSPEECH*, 2017.

[174] Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ Skerry-Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Fei Ren, Ye Jia, and Rif A. Saurous. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. *arXiv:1803.09017*, 2018.

[175] Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *J. Mach. Learn. Res.*, 15(1):949–980, January 2014.

[176] Jun Xiao and Ying Wang. A semi-fragile watermarking tolerant of laplacian sharpening. In *2008 International Conference on Computer Science and Software Engineering*. IEEE, 2008.

[177] Cihang Xie, Zhishuai Zhang, Jianyu Wang, Yuyin Zhou, Zhou Ren, and A. Yuille. Improving transferability of adversarial examples with input diversity. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2725–2734, 2019.

[178] Junichi Yamagishi, Christophe Veaux, and Kirsten MacDonald. CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92), 2019.

[179] Hengfu Yang, Xingming Sun, and Guang Sun. A semi-fragile watermarking algorithm using adaptive least significant bit substitution. *Information Technology Journal*, 2009.

[180] X. Yang, Y. Li, and S. Lyu. Exposing Deep Fakes Using Inconsistent Head Poses. In *Proc. International Conference on Acoustics, Speech and Signal Processing*, 2019.

[181] Xiaoyan Yu, Chengyou Wang, and Xiao Zhou. Review on semi-fragile watermarking algorithms for content authentication of digital images. *Future Internet*, 2017.

[182] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, and Victor Lempitsky. Few-shot adversarial learning of realistic neural talking head models. In *ICCV*, 2019.

[183] H. Zen, V. Dang, R. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu. Libritts: A corpus derived from librispeech for text-to-speech. In *Proc. Interspeech*, 2019.

[184] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech. In *INTERSPEECH*, 2019.

[185] Heiga Zen, Keiichi Tokuda, and Alan W Black. Statistical parametric speech synthesis. *Speech communication*, 51(11):1039–1064, 2009.

[186] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[187] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[188] Ru Zhang, Shiqi Dong, and Jianyi Liu. Invisible steganography via generative adversarial networks. *Multimedia tools and applications*, 2019.

[189] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Two-stream neural networks for tampered face detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1831–1839. IEEE, 2017.

[190] Wen Zhou, X. Hou, Y. Chen, Mengyun Tang, Xiangqi Huang, X. Gan, and Yong Yang. Transferable adversarial perturbations. In *ECCV*, 2018.

[191] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. Hidden: Hiding data with deep networks. In *ECCV*, 2018.