TOWARDS ROBUST AND SECURE FACE RECOGNITION: DEFENSE AGAINST PHYSICAL AND DIGITAL ATTACKS

By

Debayan Deb

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2021

ABSTRACT

TOWARDS ROBUST AND SECURE FACE RECOGNITION: DEFENSE AGAINST PHYSICAL AND DIGITAL ATTACKS

By

Debayan Deb

The accuracy, usability, and touchless acquisition of state-of-the-art automated face recognition systems (AFR) have led to their ubiquitous adoption in a plethora of domains, including mobile phone unlock, access control systems, and payment services. Despite impressive recognition performance, prevailing AFR systems remain vulnerable to the growing threat of *face attacks* which can be launched in both *physical* and *digital* domains. Face attacks can be broadly classified into three attack categories: (i) *Spoof* attacks: artifacts in the physical domain (*e.g.*, 3D masks, eye glasses, replaying videos), (ii) *Adversarial* attacks: imperceptible noises added to probes for evading AFR systems, and (iii) *Digital manipulation* attacks: entirely or partially modified photo-realistic faces using generative models. Each of these categories is composed of different attack types. For example, each spoof medium, *e.g.*, 3D mask and makeup, constitutes one attack type. Likewise, in adversarial and digital manipulation attacks, each attack model, designed by unique objectives and losses, may be considered as one attack type. Thus, the attack categories and types form a 2- layer tree structure encompassing the diverse attacks. Such a tree will inevitably grow in the future. Given the growing dissemination of "fake news" and "deepfakes", the research community and social media platforms alike are pushing towards *generalizable* defense against continuously evolving and sophisticated face attacks. In this dissertation, we first propose a set of defense methods that achieve state-of-the-art performance in detecting attack types within individual attack categories, both physical (e.g., face spoofs) and digital (e.g., adversarial faces and digital manipulation), then introduce a method for simultaneously safeguarding against each attack.

First, in an effort to impart generalizability and interpretability to face spoof detection systems, we propose a new face anti-spoofing framework specifically designed to detect unknown spoof types, namely, Self-Supervised Regional Fully Convolutional Network (*SSR-FCN*), that is trained

to learn local discriminative cues from a face image in a self-supervised manner. The proposed framework improves generalizability while maintaining the computational efficiency of holistic face anti-spoofing approaches ($<$ 4 ms on a Nvidia GTX 1080Ti GPU). The proposed method is also interpretable since it localizes which parts of the face are labeled as spoofs. Experimental results show that SSR-FCN can achieve True Detection Rate (TDR) = 65% @ 2.0% False Detection Rate (FDR) when evaluated on a dataset comprising of 13 different spoof types under unknown attacks while achieving competitive performances under standard benchmark face anti-spoofing datasets (Oulu-NPU, CASIA-MFSD, and Replay-Attack).

Next, we address the problem of defending against adversarial attacks. We first propose, *AdvFaces*, an automated adversarial face synthesis method that learns to generate minimal perturbations in the salient facial regions. Once AdvFaces is trained, it can automatically evade state-of-the-art face matchers with attack success rates as high as 97.22% and 24.30% at 0.1% FAR for obfuscation and impersonation attacks, respectively. We then propose a new self-supervised adversarial defense framework, namely *FaceGuard*, that can automatically detect, localize, and purify a wide variety of adversarial faces without utilizing pre-computed adversarial training samples. FaceGuard automatically synthesizes diverse adversarial faces, enabling a classifier to learn to distinguish them from bona fide faces. Concurrently, a purifier attempts to remove the adversarial perturbations in the image space. FaceGuard can achieve 99.81%, 98.73%, and 99.35% detection accuracies on LFW, CelebA, and FFHQ, respectively, on six unseen adversarial attack types.

Finally, we take the first steps towards safeguarding AFR systems against face attacks in *both* physical and digital domains. We propose a new **uni**fied **f**ace **a**ttack **d**etection framework, namely *UniFAD*, which automatically clusters similar attacks and employs a multi-task learning framework to learn salient features to distinguish between bona fides and coherent attack types. The proposed *UniFAD* can detect face attacks from 25 attack types across all 3 attack categories with TDR = 94.73% @ 0.2% FDR on a large fake face dataset, namely *GrandFake*. Further, *UniFAD* can identify whether attacks are adversarial, digitally manipulated, or contain spoof artifacts, with 97.37% classification accuracy.

To My Loving Family

ACKNOWLEDGMENTS

Arguably, this section has been the hardest one to write in this entire dissertation due to the sheer number of individuals who were instrumental towards the completion of my PhD research. Foremost, I would like to express my deepest gratitude towards my advisor, Dr. Anil K. Jain, for his unwavering support and encouragement to strive for excellence. Despite being a raw student at first, Dr. Jain, took a chance on me by bringing me into his research lab and patiently teaching me the rigors of scientific research. His (i) ability to systematically disseminate scientific problems, (ii) attention to details, (iii) work ethic, (iv) skill of presenting ideas in a visually-convincing manner, and (v) promptness in responding to emails, are some of the many virtues that I will always strive to instill in me. Apart from being a great scientist, he is also extraordinarily humble and kind at heart. Thank you Dr. Jain also for your friendship and guidance on matters outside of research. Thank you Dr. Ross for teaching CSE491: Selected Topics in Biometrics and admitting me into your research group prior to my PhD. Without your initial support, I would not have been writing this dissertation.

I would also like to thank my PhD committee, Dr. Arun Ross, Dr. Xiaoming Liu, and Dr. Mi Zhang, for providing valuable comments and suggestions on this dissertation. A special thanks to Dr. Liu and Dr. Ross for your patience, inspirational ideas, and suggestions during our collaborative research. Thank you Brenda Hodge, Steven Smith, Amy King, and Erin Dunlop for your administrative assistance and your patience with my extremely tardy travel reimbursement forms. A special thanks to Christopher Perry, for being a great support and help in setting up my GPU machine – without which the research conducted in this dissertation could not have been possible.

My deepest gratitude to Brendan Klare from Rank One Computing, Jianbang Zhang from Lenovo, and Biplob Debnath from NEC, for providing me with internship opportunities at their respective companies. Through the internships, I got an opportunity to view scientific research from a business perspective. I would also like the thank Ford, Lenovo, IARPA, and Facebook AI Research for their support in research conducted during my PhD program.

I am grateful for all my fellow PRIPies, especially my contemporaries (Josh, Yichun, and Sixue). Thank you Josh for your close friendship and the many beer-driven conversations we had across the world over the years. Thank you Inci for all of our conversations in the lab, your encouragements throughout, and willingness to collaborate with me on future research. A special thanks to Aishvary who has been like a brother to me for the past decade. Thank you Vishesh, Divyansh, Yichun, and Tarang for helping me gain confidence during my lows. Thank you also to Steve, Kai, Cori, Lacey, Charles, and Sunpreet. Special thanks to non-PRIPies including but not limited to Yaojie Liu, Amin Jourabloo, Joel Stehouwer, Sudipta Banerjee, Rahul Dey, Siddharth Shukla, Steven Hoffman, Adam Terwilliger, Xi Yin, and Thomas Swearingen. *Sincere apologies to all for my tardiness.*

I owe a great deal to my parents and sister for being a constant source of support and guidance while I journeyed through the ups and downs as a graduate student. Without their moral support, none of this would have been possible.

There are of course many others who have had a great impact on me along the way and I sincerely thank all of you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

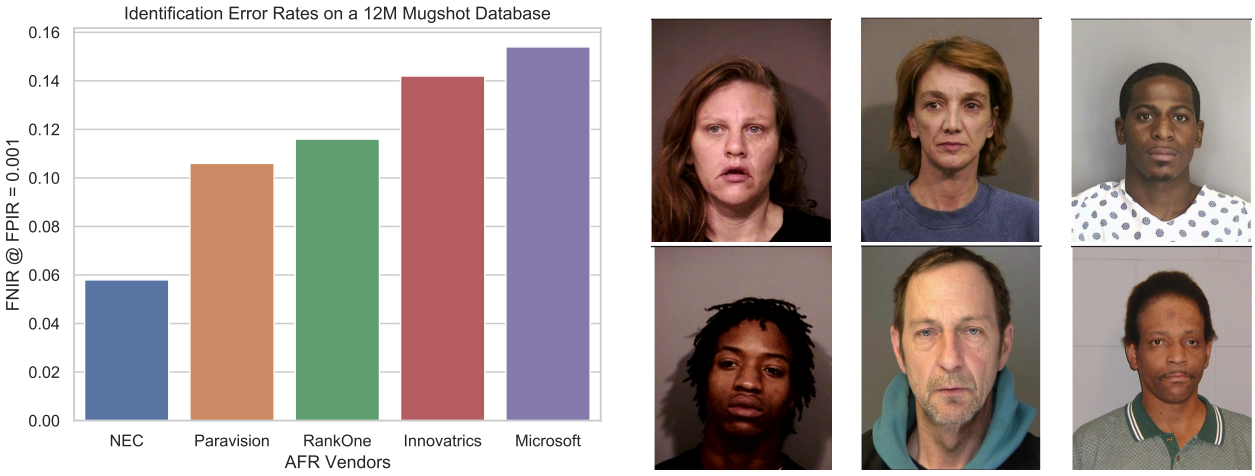
LIST OF ALGORITHMS

Chapter 1

Introduction

Automated face recognition (AFR) systems - the software that maps, analyzes, and then confirms the identity of a face in a photograph or video - is one of the most powerful surveillance tools ever made. While people interact with AFR systems as a way of unlocking their phones or sorting their photos, face recognition technologies are currently deployed in many important applications. The ubiquity of AFR systems is evident in both commercial and governmental applications. For instance, face recognition plays a vital role in identity card de-duplication to prevent a person from obtaining multiple ID cards, such as driver's licenses and passports, under different names[1]. AFR systems are also utilized by the United States Department of Homeland Security (DHS), the United States Department of Defense (DoD), along with the Federal Bureau of Investigation (FBI) and Immigration and Customs Enforcement (ICE), to determine friend or foe at security checkpoints, and to assist law enforcement officers in the field to capture face images with mobile devices, submit them to face recognition system on central servers, and quickly identify people who refuse to give their name, provide false information, or are injured and unresponsive[2]. Face recognition systems are additionally employed for surveillance and access control to secure locations. Commercial applications of automatic face recognition are also now abundant, including automatic "tag" suggestions on Facebook, organization of personal photo collections, and mobile

[1] https://bit.ly/2SO0yuL
[2] https://bit.ly/3jSO4xH



(a) (b)

Figure 1.1 (a) Identification error rates of six state-of-the-art automated face recognition vendors on a 12 million mugshot dataset, namely FRVT-2018 [22]. (b) Six mugshots representative of the FRVT-2018 dataset.

phone unlock.

The current capabilities of an AFR system depend heavily on the image acquisition conditions (*i.e.*, subject cooperation, environmental conditions, etc.). ID document verification, for instance, entails frontal-to-frontal face matching of controlled images, where faces are required to have a neutral expression, a uniform background, and controlled lighting. In these scenarios, state-of-the-art commercial off-the-shelf (COTS) face recognition systems are highly accurate and have proven to be extremely useful. Forty-three of the 50 states in the United States are utilizing AFR systems for detecting fraudulent ID documents; if the FBI needs to put a name to a face, the bureau can scan through 411 million face photos spread across state and federal databases[3]. In 2020, a large-scale face recognition evaluation, conducted by the National Institute of Standards and Technology (NIST), demonstrated that the error rates of the top performing COTS AFR systems were very low for identifying mugshot face images at rank-1 against a gallery comprising of 12 million mugshots (see Figure 1.1).

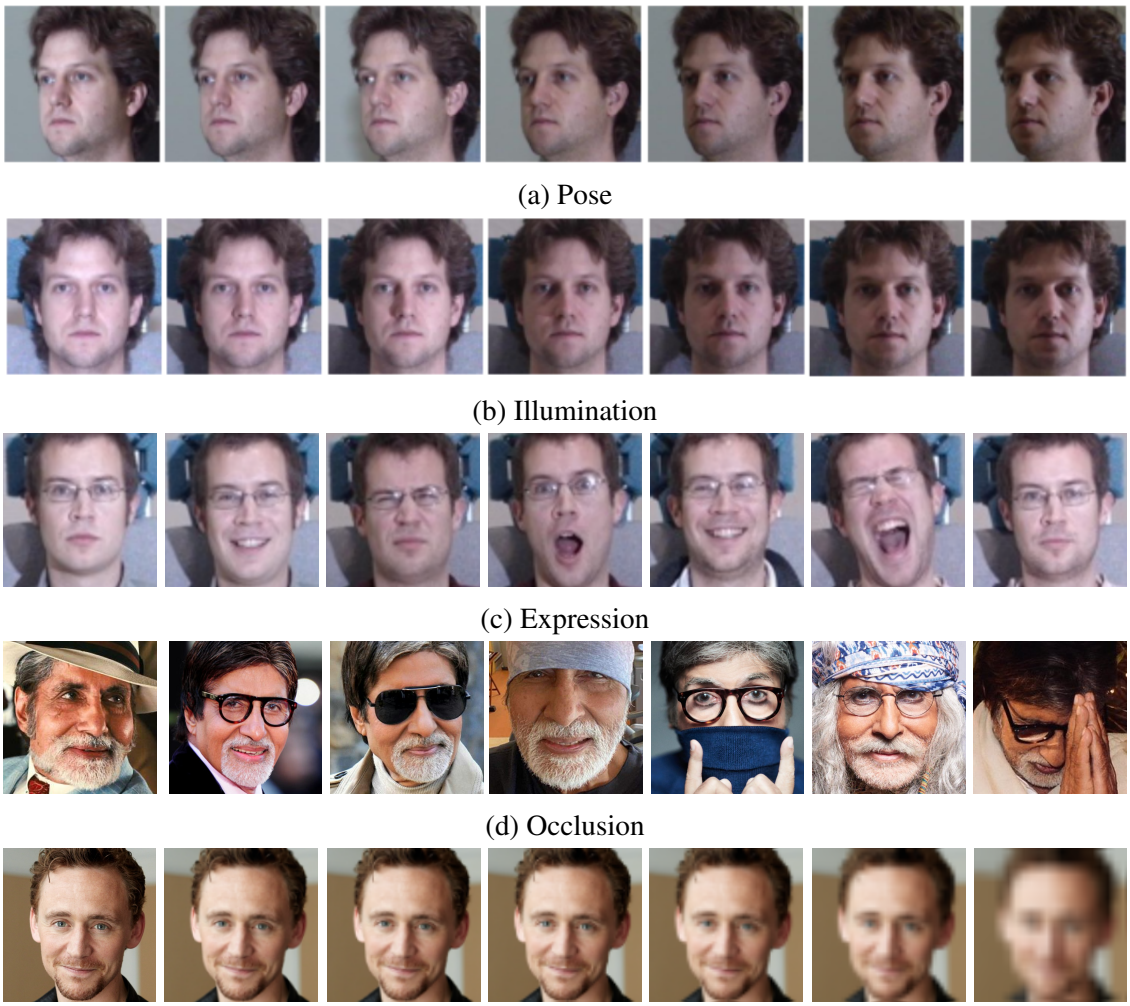
Compared to other biometric traits such as fingerprint and iris, face offers a number of advantages: (i) We, as humans, intrinsically identify others via face. Therefore, face does not reveal

[3]https://bit.ly/36edG4b

any extraneous sensitive information that people themselves do not already reveal to the public on a daily basis. Consequently, face recognition tends to be a more publicly acceptable biometric trait (compared to, say, fingerprints which are commonly associated with criminal accusations). (ii) Unlike fingerprint and iris, no specialized hardware sensors are required; digital cameras are readily available (*i.e.*, in smartphones) and are relatively inexpensive. (iii) Faces can be acquired unobtrusively at a distance, and in a covert manner, if required. (iv) Available large-scale legacy face image datasets (such as passport and driver's license) ease benchmarking face recognition performance. (v) In addition to identity, faces also reveal demographic attributes such as gender, race, and age. (vi) With the recent COVID-19 pandemic, the "touchless" nature of face image acquisition is attractive in both government and commercial applications.

With new emerging applications of AFR systems, the advantages of utilizing face biometric for identification is apparent. In today's society, smartphone and surveillance cameras are ubiquitous. As American schools, malls and offices tighten security on their premises, the number of surveillance cameras in the United States is said to grow to 85 million by 2021, compared to 70 million in 2020. China alone is expected to have 560 million networked CCTV cameras by the end of 2021[4]. The total number of surveillance cameras around the world is said to climb above 1 billion by the end of 2021[5]. With recent tragic police incidents such as the death of George Floyd in Minneapolis, Freddie Gray in Baltimore, and Breonna Taylor in Louisville, many police departments are now requiring patrol officers to wear body cameras. In this era of constant documentation of personal lives on social media websites, such as Facebook and Instagram, personal collection of face photos ("selfies") are also booming with an estimated 25,000 selfies taken by an average person (18 to 34 year olds) during their lifetime[6]. Due to this increasingly vast collection of available imagery, in addition to countless routine crimes (*e.g.*, robbery, kidnapping, assault), AFR systems are an invaluable tool for identification of persons of interest. For example, Walter Yovany-Gomez, a member of the MS-13 street gang, evaded authorities for years before FBI put

[4]https://bit.ly/3nIQt0d
[5]https://on.wsj.com/3345RvW
[6]https://bit.ly/346F0P4



(a) Pose

(b) Illumination

(c) Expression

(d) Occlusion

(e) Resolution

Figure 1.2 Sources of intra-subject variability: (a) pose, (b) illumination, and (c) expression. Each row shows intra-subject variations for the same individual in (a-c; source: PIE Dataset [23]), (d) Amitabh Bacchan (source: Google Images), and (e) Tom Hiddleston (source: Google Images).

him on its Ten Most Wanted Fugitives list in 2011. With the aid of an AFR system, investigators finally traced Mr. Gomez in 2017 via photos on Facebook profiles containing his face[7]. Without utilizing an *automated* face recognition system, manually sifting through the 250 billion photos uploaded on Facebook, till date[8], is an impossible task and a wanted criminal such as Mr. Gomez would be roaming around freely today.

Even with the successes enjoyed by AFR systems in the aforementioned scenarios, face recog-

[7] https://n.pr/3j72xWl
[8] https://bit.ly/30bGTZY

nition technology is still limited by their *unconstrained* nature of the imagery available. Accuracies of prevailing COTS AFR systems are highly sensitive to the image acquisition conditions. In unconstrained scenarios, face image acquisition is not well-controlled and subjects may be non-cooperative (or unaware). Confounding factors plaguing prevailing AFR systems include:

- **Pose:** Face poses can vary in-plane (roll) or out-of-plane rotation (yaw and/or pitch). In-plane rotations can be corrected via straightforward 2D transformations. However, out-of-plane rotations cause faces to become "self-occluded" leading to missing information and poor face recognition performance. See Figure 1.2a for some example faces with extreme pose variation.

- **Illumination:** Even for faces acquired in the natural environmental settings, lighting conditions can vary drastically (such as indoors vs. outdoors) and illumination is also affected by daily changes (such as time of day and/or weather conditions). Due to the three-dimensional nature of face, illumination across the face varies rapidly due to the shadows casted at certain angles. Illumination variation is exacerbated when the 3D face is acquired and translated into a 2D grayscale or color image. Depending on the strength of the illumination, some face features may be exaggerated or may completely vanish. Figure 1.2b shows a few examples of faces under illumination variation.

- **Expression:** In an unconstrained setting, face images may be acquired without the knowledge of the subject. Therefore, face images may be captured mid-conversation and/or while viewing something surprising, upsetting, infuriating, etc. Such deviations from a neutral (or relaxed) face expression, lead to a degradation in face recognition performance. As a result, the State Department guidelines for passport photo acquisition permits smiles but frowns on "toothy" smiles (which apparently are classified as unusual or unnatural expressions)[9]. According to the guidelines, "The subject's expression should be neutral (non-smiling) with both eyes open, and mouth closed. A smile with a closed jaw is allowed but is not preferred.".

[9]https://cbsn.ws/3i7cdiD

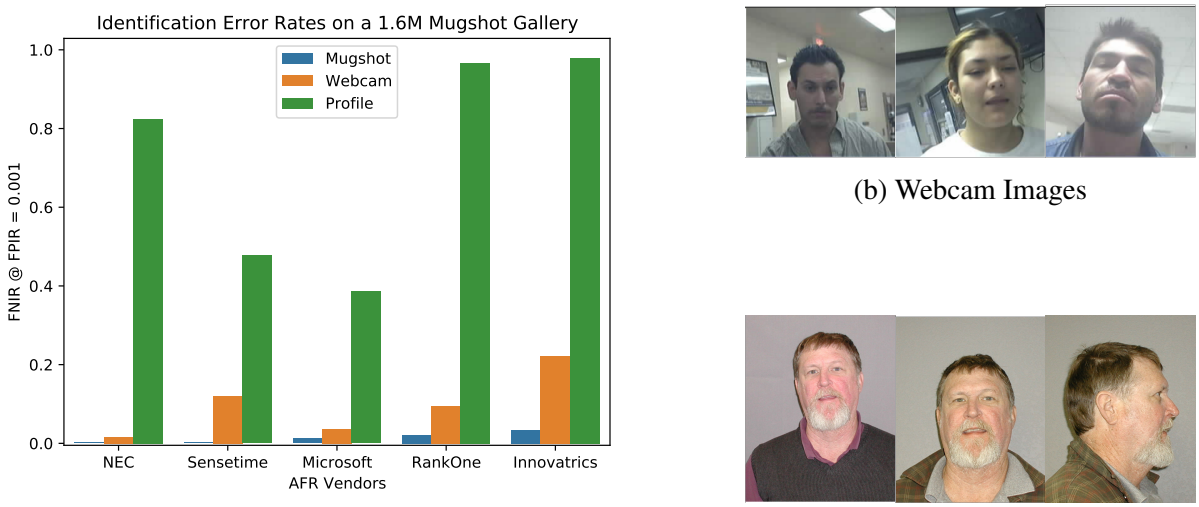
This is because smiling distorts other facial features, for example, one's eyes. Figure 1.2c shows a few examples of faces with expression changes. Extreme expression variations remain an ongoing challenge for AFR systems.

- **Occlusion:** It is quite common for unconstrained faces to contain facial accessories such as eyeglasses and sunglasses. Occluding eye regions can negatively impact face recognition performance since these facial regions are highly discriminative. Facial occlusions are problematic not only because of missing information, but also due to the extraneous and spurious information introduced. For instance, even if a person consistently wears eyeglasses, specular reflections that change based on the light source can lead to high intra-subject face variation. Figure 1.2d shows a few examples of occluded faces.

- **Aging:** Given two images of the same individual acquired multiple years apart, a robust AFR system should still be able to identify the two images as pertaining to the same individual. However, we find in Figure 1.4, that even the top performing COTS AFR systems have difficulty identifying the individual as he or she ages [37–39]. Unlike other factors, face aging is intrinsic and cannot be controlled by the subject or the acquisition environment. That is, face aging can be present in both controlled (constrained) and uncontrolled (unconstrained) scenarios.

- **Resolution:** The quality of face images severely affects face recognition performance. Figure 1.3 shows the increase in error rates when lower quality webcam face photos are matched to a mugshot gallery. In practice, unconstrained face images are of poor image quality (such as those captured from surveillance cameras). Face recognition performance on poor quality images is far from desirable and remains an ongoing challenge for the biometric community (see Figure 1.2e).

Inter-subject similarities can also lead to face recognition errors. For instance, it is challenging (even for humans) to distinguish between people with kinship relations (particularly twins, see Figure 1.5a), and also people that are not related but exhibit strong face similarities (Figure 1.5b, 1.5c).



(a)

(b) Webcam Images

(c) Profile Images

Figure 1.3 Identification error rates of six state-of-the-art automated face recognition vendors when (a) mugshots, (b) webcam images, and (c) profile faces are compared against a 1.6 million mugshot dataset (a subset of the FRVT-2018 [22]).

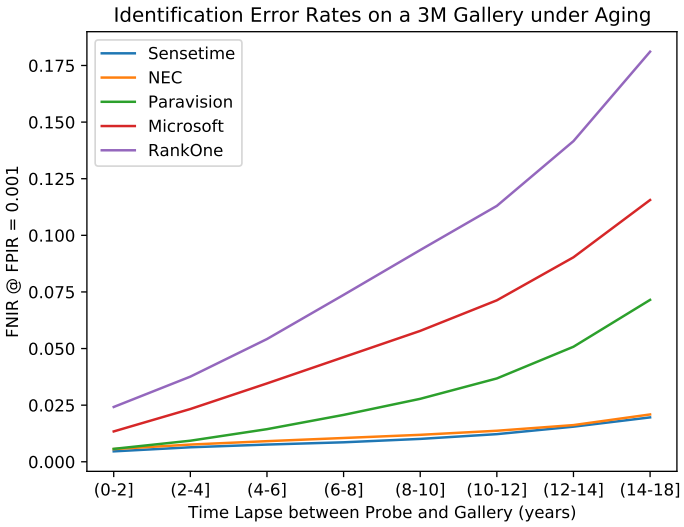
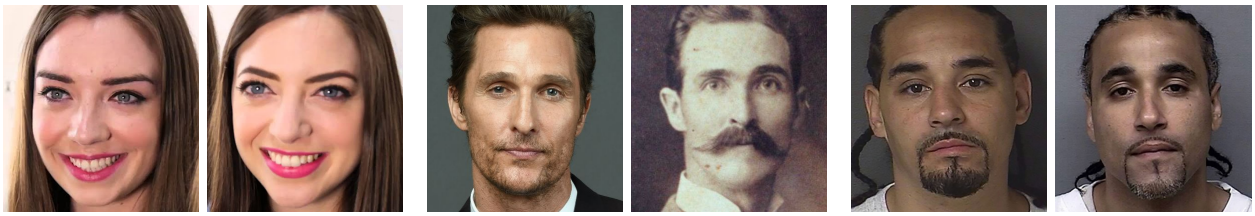


Figure 1.4 Identification error rates of six state-of-the-art automated face recognition vendors on a 3 million mugshot dataset under aging [22]. Face recognition errors increase as the time gap between a probe image and the enrolled face image in the gallery increases.

As mentioned earlier, due to the recent COVID-19 pandemic, authorities and citizens alike are enjoying the "contactless" and covert nature of face recognition. However, the covertness can also lead to its downfall with the advent of face attacks launched in both physical (such as 3D face masks) and digital domains (adversarial and digitally manipulated faces). Without the presence of a



(a) (b) (c)

Figure 1.5 Source of inter-subject similarities: (a) kinship relations (here, twins), (b) different people with no kinship relations who happen to exhibit very similar facial characters (known as "doppelgängers"), and (c) Richard Jones (right) spent 17 years in prison for a crime committed by his doppelgänger, Ricky Amos (left) (Source: https://cnn.it/2Gb1F4A).

human operator ensuring the legitimacy of face image acquisition, malicious individuals (attackers) are increasingly challenging the security of AFR pipelines for government benefits, access control, and financial gains. *The vulnerabilities of AFR systems towards face attacks and methods to defend against them will be discussed later in this chapter.*

1.1 Background

A typical AFR system may operate in various modes depending on the deployed application. In most scenarios, face images along with their identity labels are first *enrolled* as a template in a dataset, referred to as a *gallery*. Later, the AFR system takes as input an image, known as a *probe* or *query*, and matches it against one or many faces present in the gallery.

Face *verification* or *authentication* refers to a one-to-one matching scenario where the task of the AFR system is to verify whether a probe face image belongs to the individual that he/she claims to be (*e.g.*, a passport and passenger's face photo, access control, and smartphone unlock). On the other hand, face *identification* or *search* involves one-to-many comparisons in order to retrieve (from the gallery) the identity of the probe face image (whose identity is previously unknown).

In the identification scenario, the number of identities to retrieve is usually manually defined to top-k, where k depends on the application at hand. This is referred to as a *closed-set* identification scenario, where we assume that the identity of the probe is present in the gallery and we hope that the correct identity will be present in the top-k retrieved individuals. Authorities can then

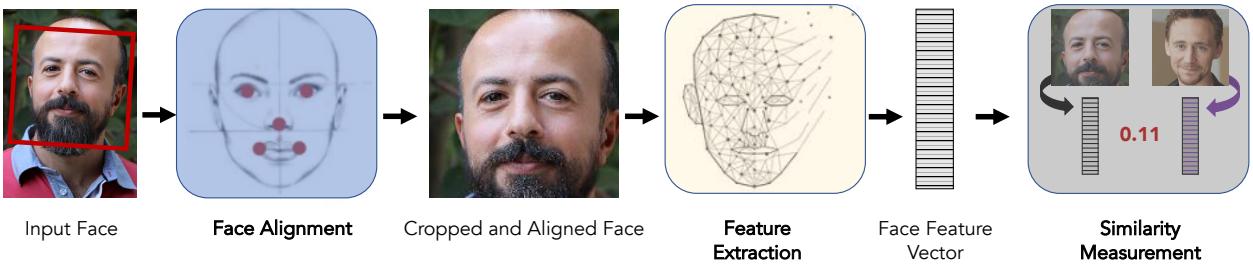


Figure 1.6 A typical Automated Face Recognition (AFR) system typically consists of (i) face detection, (ii) face alignment (to mitigate geometric distortions), (iii) feature extraction, and (iv) comparison of face representations (feature vectors).

manually review the top-k candidate retrievals to identify, say, a suspected criminal in a forensics dataset. However, real-world scenarios entail an *open-set* identification scenario, where we cannot assume that the probe's identity will indeed be present in the gallery. For open-set applications, we retrieve the top candidate matches only if they exceed a pre-defined threshold (instead of manually specifying a fixed k). This is extremely useful (for example, in watch list surveillance) since it may be impractical for authorities to manually review retrieved candidates for every probe.

1.2 Automated Face Recognition (AFR)

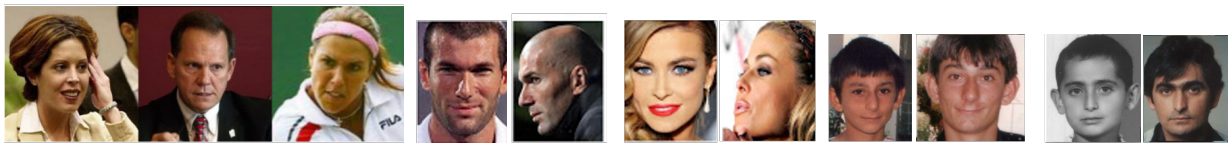
Regardless of the operating scenario (verification or identification), the primary task of all AFR systems is to compute a similarity measurement between any two face images. A robust AFR system should ideally output a high similarity measure between a face image pair of the same individual (*genuine pair*) and a low similarity measure between face image pairs of different individuals (*impostor pairs*). This process involves multiple components which all significantly contribute to the final face similarity measurement and consequently, the resulting face recognition accuracy.

1.2.1 AFR Pipeline

A typical AFR systems is composed of the following sequential modules (see Figure 1.6): (i) face detection, (ii) face alignment, (iii) face feature extraction and face representation, and (iv) similarity measurement. Each component is necessary and contributes to the overall performance

Table 1.1 Verification performance (%) under two different face detectors on LFW, CFP-FP, and AgeDB-30 [5]. Figure 1.7 shows a few examples of each of the three datasets .

Methods	**LFW [8]**	**CFP-FP [24]**	**AgeDB-30 [25]**
MTCNN+ArcFace [9]	99.83	98.37	98.15
RetinaFace+ArcFace [5]	**99.86**	**99.49**	**98.60**



(a) LFW [8] (b) CFP [24] (c) AgeDB [25]

Figure 1.7 Example faces in (a) LFW [8], (b) CFP [24], and AgeDB [25] datasets.

of the AFR system. Thus, significant attention is seen in the literature towards improving each individual component.

1.2.1.1 Face Detection

Since images may consist of a variety of different objects (including face), prior to face recognition, it is imperative that faces are spatially located in the input image. Face detection is the process of automatically (a) determining whether a face (or multiple faces) is present in an input image, and then (b) outputting the locations (2D coordinates) of all detected faces. This is a trivial task for humans, whereas, extracting face regions from arbitrary images is challenging for machines. This can likely be attributed to a high intra-class variability in the appearance of faces (*e.g.*, skin color, background noise, face pose, illumination, etc.). Failure to detect faces (missing faces in the image, or erroneously detecting non-face regions as faces) is extremely problematic, since subsequent AFR components will not add any value if they are not presented with an actual face region.

The seminal work of Viola and Jones [40] is credited to being the first real-time and accurate face detector. Since its publication in 2004, a large number of studies in literature have focused on improving face detection accuracy. Currently, MTCNN [41] and RetinaFace [5] lead the way in face detection accuracy (see Figure 1.8) and are commonly employed in face recognition litera-

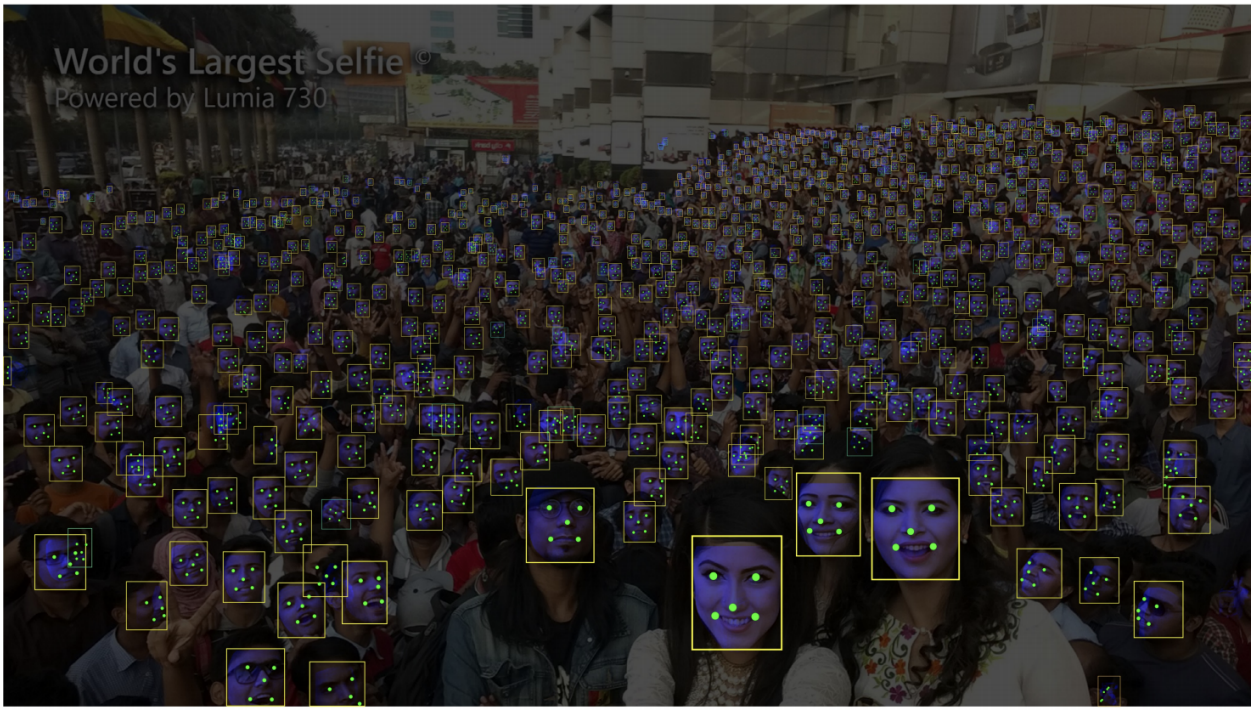
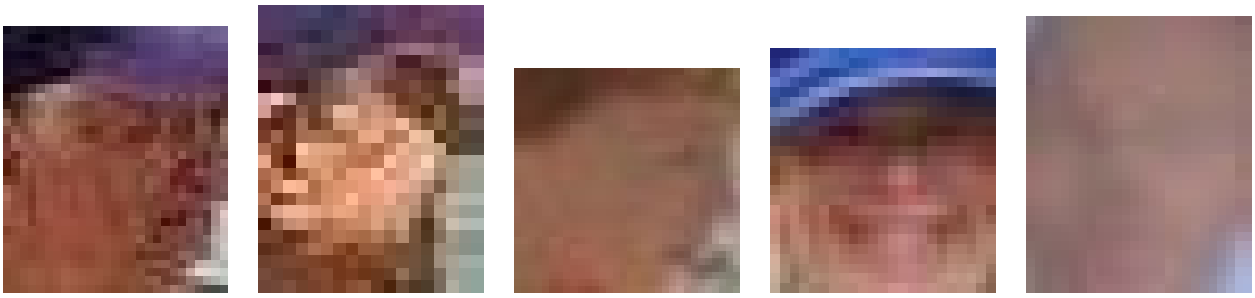


Figure 1.8 A state-of-the-art face detector, RetinaFace, can detect around 900 faces (detection threshold at 0.5) out of 1,151 people reported to be present in the "World's Largest Selfie" [5]. The yellow rectangle denotes the bounding box around a face and green dots represent the detected landmarks.



(a) (b) (c) (d) (e)

Figure 1.9 Example images that supposedly contain faces but cannot be detected by a state-of-the-art face detector, RetinaFace [5]. Note that these images are of very low resolution.

ture [6, 9, 42]. To understand the benefits of an accurate face detection system on face recognition performance, in Table 1.1 we report the face verification accuracy on three benchmark datasets, LFW [8], CFP-FP [24], AgeDB-30 [25]. Two face detectors are employed: (i) MTCNN [41] and (ii) RetinaFace [5]. We can observe an improvement in face recognition performance when a better face detector, RetinaFace, is employed. However, even with the state-of-the-art face detectors, errors can still be observed due to extreme pose variations, occlusions, and low resolution (see Figure 1.9).

1.2.1.2 Face Alignment

This step is primarily concerned with reducing intra-class variability by eliminating geometric distortions present in the face regions. In particular, geometric and photographic variations may be mitigated by transforming the detected face regions to a canonical view (front face view). Face alignment aims at determining correspondences between face images based on landmarks/fiducial points (*e.g.*, eyes, nose, mouth, chin, jaw, etc.). The most straightforward alignment technique is a simple 2D rigid affine transformation based on two eye locations to account for face size and in-plane head rotation [6,43]. More sophisticated alignment methods involve employing 3D modeling techniques to "frontalize" the face, which also accounts for out-of-plane head rotations. However, such 3D face alignment techniques are generally time-consuming and increases the overhead time associated with an AFR system.

Face alignment requires a set of pre-defined reference points (*i.e.*, landmarks) which are points defining a base face position (*i.e.*, position of eyes, nose, and mouth for an average person). A common approach for acquiring reference points is to compute the average landmark points extracted from a large face dataset. After obtaining reference points, for transforming a probe face in the 2D space, we have four possible types of transformations (see Figure 1.10):

- **Euclidean Transformation** which is a rigid transformation preserving distances between each pair of points. The Euclidean transformation allows rotating and translating the face (3 Degrees of Freedom).

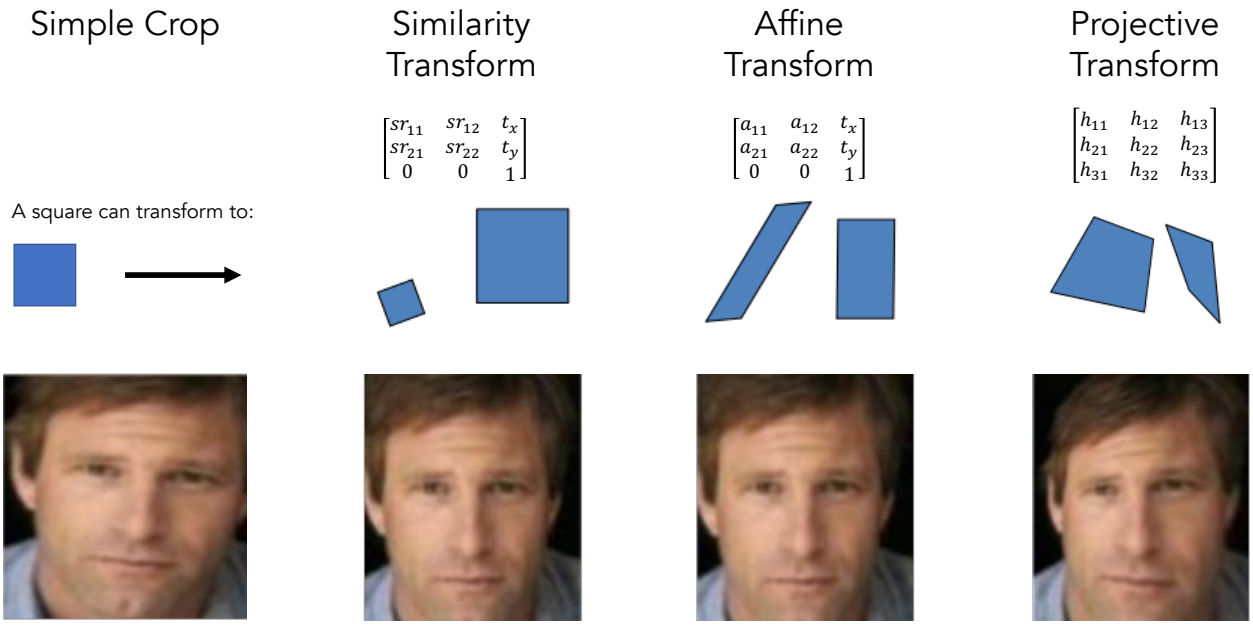


Figure 1.10 Illustration of various 2D face alignment techniques: (i) simply cropping the face region, (ii) similarity transform (scale and rotation), (iii) affine transformation (rotation, scaling, and shear mapping), and (iv) projective transformation (perspective deformation) [7].

- **Similarity Transformation** also allows for scaling (making faces bigger/smaller) and therefore, does not preserve distances between points (4 Degrees of Freedom).

- **Affine Transformation** includes point, straight line, and plane preservation. In total, affine transformation allows translation, rotation, scaling, and changing aspect ratio and shear mapping (6 Degrees of Freedom).

- **Projective Transformation** is the most advanced form of 2D transformation. Every parameter in the transformation matrix is independent of each other (8 Degrees of Freedom).

From Figure 1.10, we find that 2D transformations with higher degrees of freedom have lower projection error rates (MSE distance to the reference points) but yield unnatural-looking faces. An interesting question arises, *"What is more beneficial to AFR systems, low projection error or natural looking faces?"* To answer this, while keeping the entire AFR pipeline consistent and only changing the face alignment method, the verification accuracy on LFW dataset [8] is reported in Table 1.2. We find that more natural looking faces (similarity transform) are better than allowing for more degrees of freedom (affine, projective). Consequently, the most commonly employed face

Table 1.2 Verification Performance (%) of FaceNet [6] AFR system under different face alignment techniques [7].

Methods	**LFW Protocol [8]**	**BLUFR Protocol @ FAR=0.1%**
SimpleCrop+FaceNet	97.9	85.59
Similarity+FaceNet	**98.1**	**88.42**
Affine+FaceNet	97.9	86.47
Projective+FaceNet	97.7	85.63

Euclidean transform is excluded as it assumes reference points are scale-invariant.

Table 1.3 Verification Performance (%) on LFW [8] for different face feature extractors.

Methods	**Year**	**Face Representation**	**LFW [8]**
Eigenfaces [45]	1991	Holistic	60.02
Fisherfaces [46]	1997	Holistic	87.47
HighDim-LBP [47]	2013	Local	95.17
Joint Bayesian [48]	2012	Local	96.33
DeepFace [49]	2014	Learned	97.35
DeepID [50]	2014	Learned	97.45
VGGFace [51]	2015	Learned	98.95
FaceNet [6]	2015	Learned	99.63
SphereFace [44]	2017	Learned	99.42
CosFace [42]	2018	Learned	99.73
ArcFace [9]	2019	Learned	99.83

alignment method in literature involves (a) detecting 5 landmark points (2 eyes, nose, and 2 mouth corners) and subsequently, (b) applying a similarity transformation [42, 44].

1.2.1.3 Feature Extraction

After obtaining a cropped and aligned face image, the next module in an AFR system involves extracting a set of numerical values (known as a *feature vector* or a *representation*) that best describes the input face. The simplest set of features are the raw pixel values of the input face. However,

such features may include spurious and irrelevant information (such as the colors present in non-face regions). Instead, matching results can be enhanced if we devise a method for extracting both high-level features (distances between facial components and their relative locations and ratios) and low-level features (wrinkles and face marks and scars). However, a feature vector must be carefully constructed such that we do not have unnecessarily large (and likely redundant) features which may negatively impact recognition rates. Table 1.3 provides a summary of a few studies primarily focused on improving face feature extraction (some of which will be discussed in Section 1.3.1).

1.2.1.4 Similarity Measurement

The final task of an AFR system is to compute a measure of similarity between face representations. The most obvious choice is Euclidean distance between feature vectors, however, other distance metrics such as cosine, Manhattan, histogram intersection, log-likelihood statistics, chi-square statistics, etc., may also improve recognition performance. In practice, cosine similarity between feature vectors is the most widely adopted similarity metric in literature [6, 9, 42, 44].

1.3 Evolution of Face Recognition

Humans have been drawn to the idea of identifying criminals based on facial characteristics since as long as the $19th$ century. Based on anthropometric measurements, Alphonse Bertillon devised a method of identifying and tracking criminals in 1879 [52]. The U.S. adopted the Bertillon system in 1887 which was later replaced by fingerprinting in the early $20th$ century. However, the concept of utilizing face photos of criminals (known as *mugshots*) for identification is still used worldwide.

The earliest pioneers of automating face recognition were Woodrow Bledsoe, Helen Wolf, and Charles Bisson[10]. In 1964 and 1965, Bledsoe, along with Wolf and Bisson began work using computers to recognize the human face. Since their work was funded from an unnamed intelligence

[10]https://bit.ly/30exJf5

agency, much of their work was never published. However, we do know that their initial work involved the manual marking of various "landmarks" on the face such as eye centers, mouth corners, etc., which were then mathematically rotated by a computer to compensate for pose variation (face alignment). Afterwards, the distances between landmarks were automatically computed and compared between images to determine the identity (similarity measurement). In other words, following the AFR pipeline outlined earlier, Bledsoe and his peers utilized face landmarks and their distances and ratio as the face representation. This work is deemed to be crucial since it was the first to show face as a valuable biometric trait for person identification.

Since Bledsoe's efforts on devising an automated face recognition system, researchers have devoted the past 50+ years on improving each stage of the AFR pipeline.

1.3.1 Face Representations

The evolution of AFR systems can be roughly categorized into three feature representation approaches: (i) holistic, (ii) local, and (iii) learned representations (see Table 1.3).

1.3.1.1 Holistic Face Representations

The first fully automated face recognition utilized *holistic face representations* where all pixels in the input face image are used to derive the face represenation. These studies rely heavily on accurate face alignment (typically using eye locations) which is challenging when non-frontal faces are encountered. Eigenfaces [45] was among the first AFR system to utilize holistic face features. A low-dimensional "face-space" is computed for a training dataset comprised of N unlabeled face images using Principal Component Analysis (PCA). The "face-space" is a set of $M < N$ eigenvectors corresponding to the largest M eigenvalues of the covariance matrix of the N training images. Eigenfaces was later extended to Fisherface [46] via supervised dimensionality reduction (Linear Discriminant Analysis) to find a subspace that minimizes intra-person variability and maximizes inter-person similarity. Holistic methods do not generalize well to other datasets not encountered during training (see Table 1.3).

1.3.1.2 Local Face Representations

Face features can also be extracted from overlapping patches in the face image at multiple scales to form local face representations. To incorporate holistic information, local features can be concatenated into a final feature vector describing the input face image. Typically, the final face representation is often over-complete with high-dimensionality (full of redundant information). Feature selection (*e.g.*, boosting) or dimensionality reduction (*e.g.*, PCA, LDA) are adopted to achieve a more compact face representation. Ahonen *et al.* proposed utilizing Local Binary Patterns (LBP) for face recognition. In order to utilize both local and holostic facial charecteristics, they divide the face image into a grid. A histogram of LBP features is computed for each cell in the grid and resulting face representation comprises a concatenated histogram. High-dimensional features sampled at multiple scales with a Joint Bayesian classifier [48] also achieves impressive face recognition performance on LFW (see Table 1.3).

1.3.1.3 Learned Face Representations

Due to the advent of large-scale face datasets and cheaper and effecient computational resources (*e.g.*, GPUs), the recent decade has witnessed major efforts towards automatic feature extraction algorithms powered by Convolutional Neural Networks (CNNs). In contrast to the manually designed ("handcrafted") holistic and local representations, face representations can be automatically learned by CNNs from large-scale face datasets. Such data-driven methods have almost completely replaced traditional methods in face recognition literature [6, 9, 42, 44, 49, 50]. In addition to state-of-the-art face recognition accuracy compared to traditional methods (see Table 1.3), CNN-based AFR systems offer a fixed-length, compact, and highly discriminative feature vector. The dimensionality of the feature representation is hierarchically reduced to convolutional and pooling layers (both local and holistic representations are jointly learned). Typically, the output of the final layer consists of a D-dimensional feature vector (where D is a manually adjudicated hyper-parameter). After the seminal work on CNN-based AFR systems such as DeepFace [49] and DeepID [50] in 2014, studies in the subsequent years have primarily focused on better objective functions (also

known as "loss functions") for training CNNs with improved discrimination power of the feature representation. Wen *et al.* proposed a center loss function which aims to minimize the intra-subject variation by minimizing the distance between feature vectors pertaining to the same identity [53].

1.4 Benchmarking AFR Systems

1.4.1 Evaluation Metrics

As mentioned earlier, AFR systems may operate under two scenarios: (i) face verification and (ii) face identification.

An AFR system is tasked to determine whether a pair of face images belong to the same subject via a decision threshold on the similarity measure (similarity score). Two commonly adopted evaluation metrics for face verification are (i) verification accuracy (or rate) and (ii) True Accept Rate (TAR) at a fixed False Accept Rate (FAR).

$$\text{Verification Accuracy}(\tau) = \frac{\text{Number of successful pairwise matches}}{\text{Total number of image pairs}} \tag{1.4.1}$$

A match is deemed successful for an image pair (x,y) if: (i) $similarity(x, y) > \tau$ when x and y are two faces belonging to the same person (genuine pair), and (ii) $similarity(x, y) <= \tau$ when x and y are two faces belonging to two different people (impostor pair). Here, the threshold τ is determined via cross-validation. Verification accuracy is commonly utilized in the LFW protocol [8] where the number of genuine and impostor pairs are balanced.

In real-world scenarios, AFR systems operate at a pre-determined match threshold (τ). Typically, we report the True Accept Rate at a pre-determined False Accept Rate, say, 0.1%. The τ is

determined via an Receiver Operating Characteristic (ROC) curve. Formally,

$$TAR(\tau) = \frac{\text{Number of genuine pairs with similarity score} > \tau}{\text{Total number of genuine pairs}}$$
$$FAR(\tau) = \frac{\text{Number of impostor pairs with similarity score} > \tau}{\text{Total number of impostor pairs}}$$

In the case of face identification, the AFR system is presented with a gallery of face images (known identities). Then, given a probe image, the AFR system needs to determine the identity of the probe from the gallery. In closed-set identification, we assume the identity of the probe is present in the gallery. Therefore, we simply need to determine the rank at which the true mate is retrieved (say at K) out of a gallery size. Then the closed-set accuracy can be computed via,

$$\text{Retrieval Rate}(K) = \frac{\text{Number of successfully retrieved probes within K retrievals}}{\text{Total number of probes}}$$

Retrieval Rate(K) is commonly referred to as Rank-K accuracy. For example, Rank-1 accuracy refers to the accuracy with which an AFR system successfully retrieves the correct identify as the first entry in a list of potential matches ranked in descending order via similarity scores with the probe image.

In the open-set identification scenario, the system needs to first determine whether the probe is in the gallery prior to retreival. In this case, the commonly employed evaluation metric is True Positive Identification Rate (TPIR) at False Positive Identification Rate (FPIR):

$$TPIR(\tau) = \frac{\text{Number of successfully retrieved mated probes at Rank-1}}{\text{Total number of mated probe}}$$
$$FPIR(\tau) = \frac{\text{Number of falsely retrieved non-mates at Rank-1}}{\text{Total number of non-mated pairs}}$$

Here, a successfully retrieved mated probe means that the true mate is returned as the top-1 result and the similarity score between probe and true mate is greater than τ.

1.4.2 Face Datasets

All efforts in designing a state-of-the-art face recognition system would have been in vain without advancements in acquiring more and more challenging face datasets for benchmarking AFR systems. Indeed, due to privacy concerns, many studies evaluate their proposed methods on face datasets acquired in-house. However, face recognition performance today would have been far from desirable had it not been for the public release of large-scale face datasets.

FERET [26] and FRGC [26] are among the first datasets utilized for benchmarking face recognition performance (see Figure 1.11). These datasets greatly contributed to advancements in AFR systems. However, these datasets were acquired under controlled conditions and were mainly geared towards studying specific challenges associated with face recognition (*e.g.*, pose, illumination, and expression). These datasets were extremely valuable for evaluating AFR performance under images exhibiting specific challenges, but were not very representative of face images encountered in the real-world.

Huang *et al.* released the Labeled Faces in the Wild (LFW) dataset which was acquired by scraping the Internet for celebrity face photos. The LFW dataset includes 13,233 face images of 5,749 different people. Face images were automatically detected via the Viola-Jones face detector [40]. Huang *et al.* also proposed the LFW protocol for benchmarking AFR accuracy: 10-fold cross-validation on face verification of 300 genuine pairs and 300 impostor pairs.

1.4.3 Constrained Face Recognition

The National Institute of Standards and Technology (NIST) began benchmarking the accuracy of face recognition systems in 1993 with the FERET program [26]. Since then, commercial face recognition systems have been evaluated in multiple Face Recognition Vendor Tests (FRVTs). Table 1.4 documents state-of-the-art constrained face recognition performance since 1993 till date. With the exception of webcam and profile face images (see Figure 1.3), most of the evaluations in the Ongoing FRVT are performed on a constrained face dataset acquired by NIST. These benchmarks are invaluable for providing comparing state-of-the-art AFR algorithms. NIST has access

Table 1.4 Benchmarking AFR performance throughout the years in NIST evaluations on frontal and constrained faces.

Study	Year	Gallery Size	Rank-1 Accuracy (%)	TAR (%) @ 0.1% FAR
FERET [26]	1993-94	316	78	21
FERET [26]	1996-97	831	95	46
FRVT [54]	2002	37,437	73	80
FRGC [26]	2005	16,028	N/A	99
FRVT [54]	2006	N/A	N/A	99
MBE [55]	2010	1.6M	92	99
FRVT [56]	2014	1.6M	96	N/A
FRVT [22]	Ongoing	12M	99.98	99.99

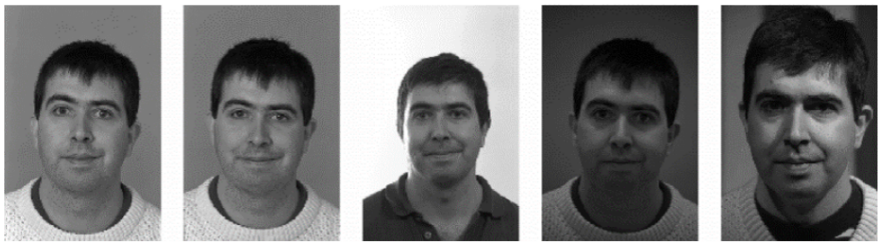
to large operational databases and conducts extensive testing of multiple algorithms on protocols that mimic operational scenarios[11].

1.4.4 Unconstrained Face Recognition

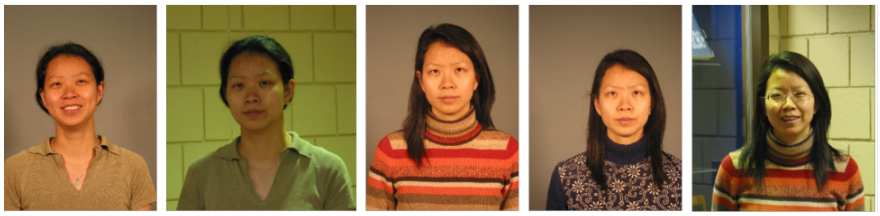
Prevailing state-of-the-art methods for unconstrained face recognition have been benchmarked by the LFW [8] database protocol since its release in 2007. Recent deep learning based AFR systems achieve accuracies above 99% via the LFW protocol (*e.g.*, FaceNet [6], CosFace [42], SphereFace [44], ArcFace [9]). As mentioned earlier, a major reason these data-driven methods are so successful is due to the availability of large-scale training datasets. Here, we enumerate a few of the training datasets that are commonly employed for training state-of-the-art face recognition systems:

- CASIA-WebFace [10] consists of 0.5M face images of 10,575 celebrities acquired "in-the-wild" by scraping the Internet.
- MS-Celeb-1M [28] contains 8M face images of 85K subjects. Similar to CASIA-WebFace, MS-Celeb also contains celebrity face photographs collected from the Internet.

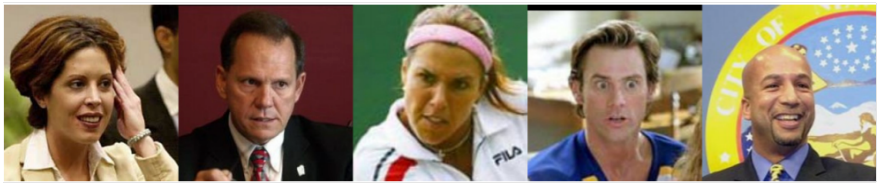
[11] http://www.nist.gov/itl/iad/ig/face.cfm



(a) FERET [26]



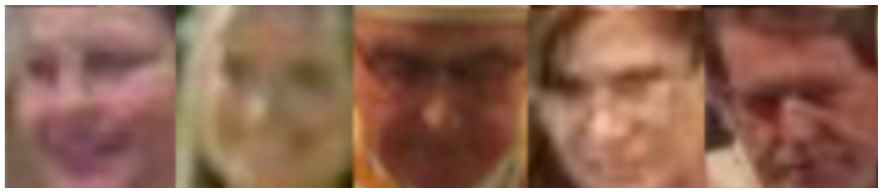
(b) FRGC [26]



(c) LFW [8]

(d) IJB-A [27]

(e) MS-Celeb-1M [28]



(f) TinyFace [29]

Figure 1.11 Example face images from (a) FERET [26], (b) FRGC [26], (c) LFW [8], (d) IJB-A [27], (e) MS-Celeb-1M [28], and (f) TinyFace [29]. Datasets (a) and (b) contain face images under relatively controlled acquisition conditions. Datasets (c-f) contain more unconstrained face images (*e.g.*, collected from the Internet).

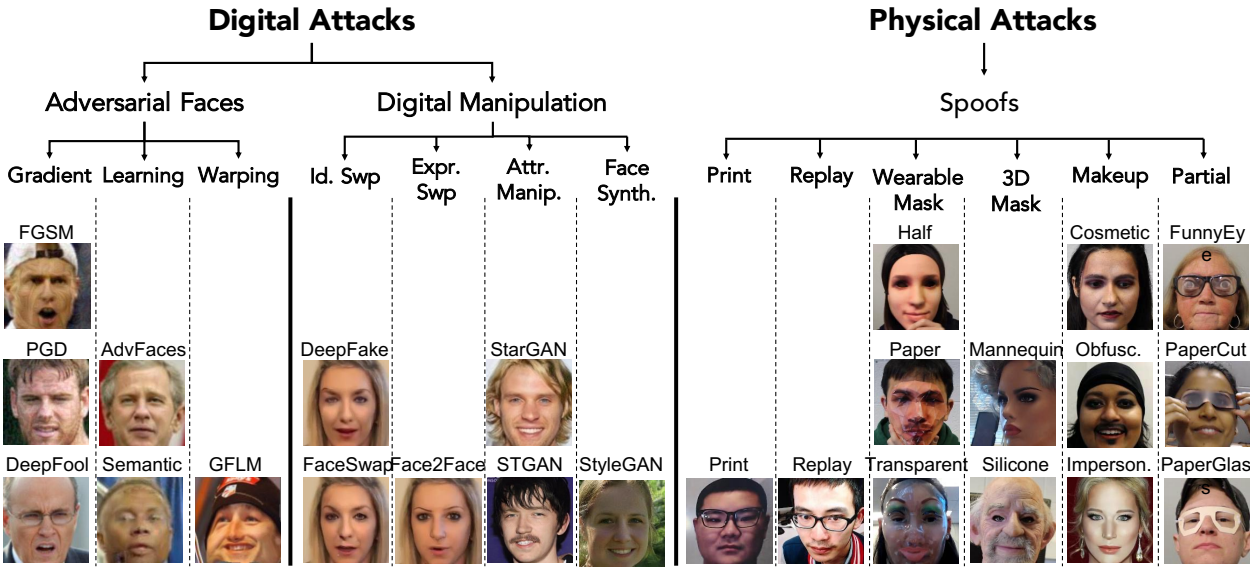


Figure 1.12 Face attacks against AFR systems are continuously evolving in both digital and physical spaces. Given the diversity of the face attacks, prevailing methods fall short in detecting attacks across all three categories (*i.e.*, adversarial, digital manipulation, and spoofs).

- VGGFace2 [57] consists of 3.3M celebrity face photos from 9K identities with 362 images per subject on average.

Example face images from some of these datasets are shown in Figure 1.11.

1.5 Vulnerabilities of AFR Systems

As mentioned earlier, the accuracy, usability, and touchless acquisition of state-of-the-art AFR systems have led to their ubiquitous adoption in a plethora of domains, including mobile phone unlock, access control systems, and payment services. Despite this impressive recognition performance, current AFR systems remain vulnerable to the growing threat of face attacks both in physical and digital domains.

For instance, an attacker can hide his identity by wearing a 3D mask [58], or intruders can assume a victim's identity by digitally swapping their face with the victim's face image [20]. With unrestricted access to the rapid proliferation of face images on social media platforms, launching attacks against AFR systems has become even more accessible. Given the growing dissemination

of "fake news" and "deepfakes" [59], the research community and social media platforms alike are pushing towards *generalizable* defense against continuously evolving and sophisticated face attacks.

In literature, face attacks can be broadly classified into three attack categories: (i) Spoof attacks: artifacts in the *physical* domain (*e.g.*, 3D masks, eye glasses, replaying videos) [1], (ii) Adversarial attacks: imperceptible noises added to probes for evading AFR systems [60], and (iii) Digital manipulation attacks: entirely or partially modified photo-realistic faces using generative models [20]. Within each of these categories, there are different attack types. For example, each spoof medium, *e.g.*, 3D mask and makeup, constitutes one attack type, and there are 13 common types of spoof attacks [1]. Likewise, in adversarial and digital manipulation attacks, each attack model, designed by unique objectives and losses, may be considered as one attack type. Thus, the attack categories and types form a 2-layer tree structure encompassing the diverse attacks (see Figure 1.12). Such a tree will inevitably grow in the future.

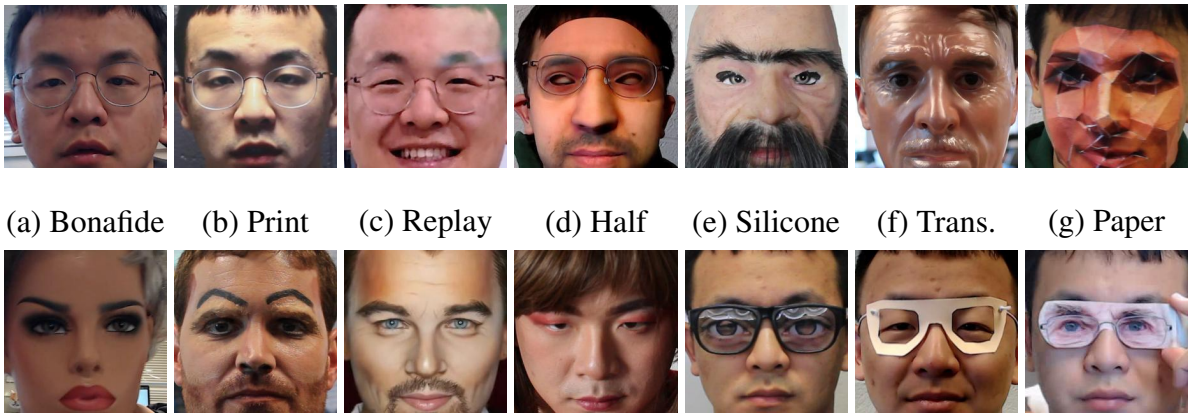
In order to safeguard AFR systems against these attacks, numerous face attack detection approaches have been proposed [20, 21, 61–63]. Despite impressive detection rates, prevailing research efforts focus on a few attack types within *one* of the three attack categories. Since the exact type of face attack may not be known *a priori*, a generalizable detector that can defend an AFR system against any of the three attack categories is of utmost importance.

1.5.1 Physical Face Spoofs

Face presentation attacks[12] are "physical fake faces" which can be constructed with a variety of different instruments (presentation attack instruments), *e.g.*, 3D printed masks, printed paper, or digital devices (video replay attacks from a mobile phone) with a goal of enabling an attacker to impersonate a victim's identity, or alternatively, obfuscate their own identity (see Figure 1.13). With the rapid proliferation of face images/videos on the Internet (especially on social media web-

[12]ISO standard IEC 30107-1:2016(E) defines presentation attacks as *"presentation to the biometric data capture subsystem with the goal of interfering with the operation of the biometric system"* [64]

sites, such as Facebook, Twitter, or LinkedIn), replaying videos containing the victim's face or presenting a printed photograph of the victim to the AFR system is a trivial task [65]. Even if a face presentation attack detection system could trivially detect printed photographs and replay video attacks (*e.g.*, with depth sensors), attackers can still attempt to launch more sophisticated attacks such as 3D masks [66], make-up, or even virtual reality [67].



(a) Bonafide (b) Print (c) Replay (d) Half (e) Silicone (f) Trans. (g) Paper

(h) Mann. (i) Obf. (j) Imp. (k) Cosm. (l) Funny (m) PaperG (n) PaperC

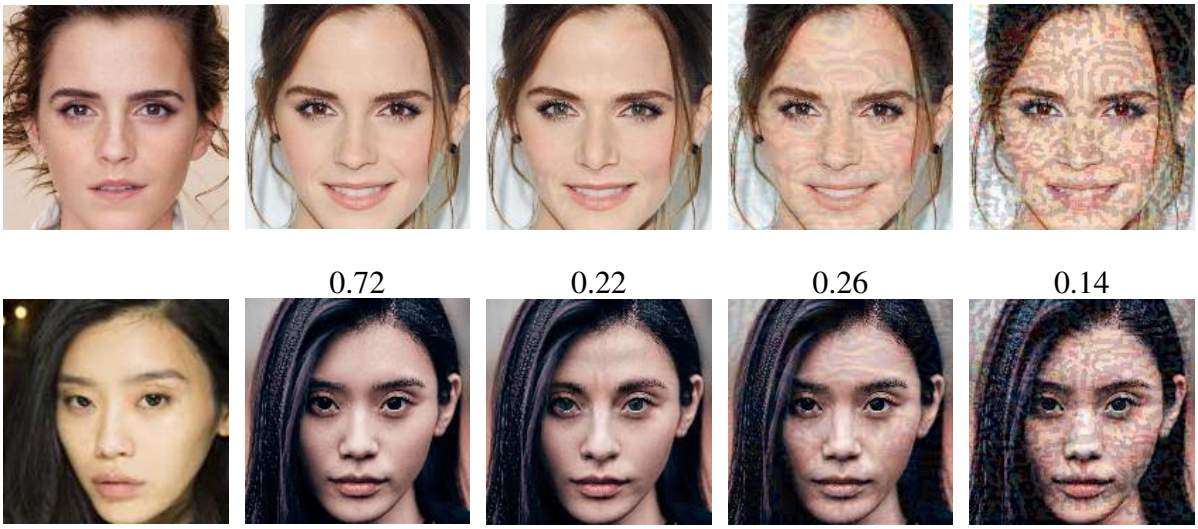
Figure 1.13 Example presentation attacks: Simple attacks include (b) printed photograph, or (c) replaying the victim's video. More advanced presentation attacks can also be leveraged such as (d-h) 3D masks, (i-k) make-up attacks, or (l-n) partial attacks [30]. A bonafide face is shown in (a) for comparison. Here, the presentation attacks in (b-c, k-n) belong to the same person in (a).

The need for preventing face attacks is becoming increasingly urgent due to the user's privacy concerns associated with spoofed systems. Failure to detect face attacks can be a major security threat due to the widespread adoption of automated face recognition systems for border control [68]. Indeed, with the advent of Apple's iPhone X and Samsung's Galaxy S8, all of us are carrying automated face recognition systems in our pockets embedded in our smartphones. Face recognition on our phones facilitates (i) unlocking the device, (ii) conducting financial transactions, and (iii) access to privileged content stored on the device.

1.5.2 Digital Adversarial Faces

From Table 1.3 we saw that prevailing AFR systems are based on automatic feature extraction methods powered by CNNs. However, CNN models have been shown to be vulnerable to *adver-*

sarial perturbations[13] [69–72]. Szegedy *et al.* first showed the dangers of *adversarial examples* in the image classification domain, where perturbing the pixels in the input image can cause CNNs to misclassify the image even when the amount of perturbation is imperceptible to the human eye [69]. Despite impressive recognition performance, prevailing AFR systems are still vulnerable to the growing threat of adversarial examples (see Figure 1.14).



(a) Enrolled Face (b) Input Probe (c) AdvFaces (d) PGD [33] (e) FGSM [70]

Figure 1.14 Example gallery and probe face images and corresponding synthesized adversarial examples. (a) Two celebrities' real face photo enrolled in the gallery and (b) the same subject's probe image; (c) Adversarial examples generated from (b) by our proposed synthesis method, AdvFaces; (d-e) Results from two adversarial example generation methods. Cosine similarity scores ($\in [-1, 1]$) obtained by comparing (b-e) to the enrolled image in the gallery via ArcFace [9] are shown below the images. A score above **0.28** (threshold @ 0.1% False Accept Rate) indicates that two face images belong to the same subject. Here, a successful obfuscation attack would mean that humans can identify the adversarial probes and enrolled faces as belonging to the same identity but an automated face recognition system considers them to be from different subjects.

To attack an AFR system, a hacker can maliciously perturb his face image in a manner that can cause AFR systems to match it to a target victim (*impersonation attack*) or any identity other than the hacker (*obfuscation attack*). Yet to the human observer, this adversarial face image should appear as a legitimate face photo of the attacker (see Figure 3.2d). This is different from face *pre-*

[13] Adversarial perturbations refer to altering an input image instance with small, human imperceptible changes in a manner that can evade CNN models.

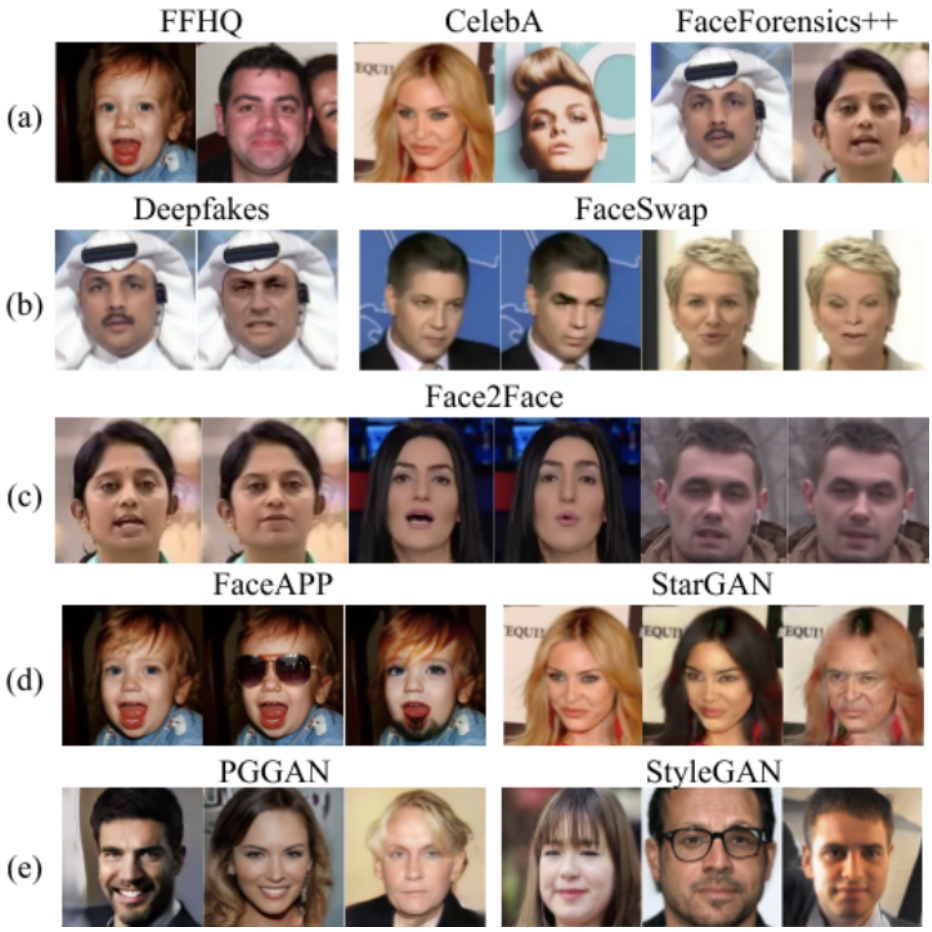


Figure 1.15 Examples of digitally manipulated faces. (a) Real images/frames from FFHQ, CelebA and FaceForensics++ datasets; (b) Paired face identity swap images from FaceForensics++ dataset; (c) Paired face expression swap images from FaceForensics++ dataset; (d) Attributes manipulated examples by FaceAPP and StarGAN; (e) Entire synthesized faces by PGGAN and StyleGAN. Collage sourced from [20].

sentation attacks, where the hacker assumes the identity of a target by presenting a physical fake face to the AFR system (see Figure 3.2). However, in the case of presentation attacks, the hacker needs to actively participate by wearing a mask or replaying a photograph/video of the genuine individual which may be conspicuous in scenarios where human operators are involved (such as airports). On the other hand, adversarial faces, do not require active participation of the subject during authentication (comparison between adversarial probe and gallery images).

Consider, for example, the United States Customs and Border Protection (CBP), the largest federal law enforcement agency in the United States [73], which (i) processes entry to the country

for over a *million* travellers *everyday* [74] and (ii) employs automated face recognition for verifying travelers' identities [75]. To evade being identified as an individual in a CBP watchlist, a terrorist can maliciously enroll an adversarial image in the gallery such that upon entering the border, his legitimate face image will be matched to a known and benign individual or to a fake identity previously enrolled in the gallery.

1.5.3 Digital Face Manipulation

Digital manipulation attacks, made feasible by Variational AutoEncoders (VAEs) and Generative Adversarial Networks (GANs), can generate entirely or partially modified photorealistic face images [20] (see Figure 1.15). Digital manipulation attack types can be broadly classified into the following:

Identity Swapping: These methods digitally replace the face of one person with the face of another person. For instance, *FaceSwap* [76] inserts famous actors into movie clips in which they never appeared. *DeepFakes* also performs face swapping via deep learning algorithms.

Expression Swapping: Expressions in a face image can be digitally and artificially replaced with another [77]. These methods swap expressions in real-time with only RGB cameras.

Attribute Manipulation Utilizing state-of-the-art GANs, studies such as StarGAN [78] and ST-GAN [79] focus on attribute manipulation by altering single or multiple attributes in a face image, *e.g.*, gender, age, skin color, hair, and glasses.

Entire Face Synthesis: Powered by large-scale high-resolution face datasets and the prevelance of GANs, an attacker can easily synthesize entire face images of unknown identities, whose realism is such that even humans have difficulty assessing if it is genuine or manipulated [80].

1.6 Dissertation Contributions

Automated face recognition has been studied extensively for more than four decades. Significant improvements have been made progressively in each component (face detection, alignment, feature extraction, matching) in order to build a highly discriminative and robust AFR system, at least for constrained and semi-constrained face recognition. However, as the technology becomes more widely adopted, safeguarding AFR systems against continuously evolving face attacks in both physical and digital domains is of the utmost importance. Since the exact type of face attack may not be known *a priori*, the need for a generalizable detector that can defend an AFR system against any of the three attack categories (spoofs, adversarial, and digital manipulation) is evident.

This dissertation first focuses on designing state-of-the-art defense methods to safeguard AFR systems against individual attack categories. Lastly, we propose a new framework to defend AFR systems against both physical and digital attacks.

The main contributions of this dissertation are as follows:

1. A generalizable, interpretable, and accurate face anti-spoofing method for detecting physical fake faces (such as 3D masks) in order to mitigate physical spoof attack on state-of-the-art AFR systems.

2. An automatic adversarial face synthesis method for generating digital fake faces that can impersonate a victim or obfuscate one's identity by evading state-of-the-art AFR systems. With a powerful adversarial face synthesizer, we can further investigate robustness of prevailing face recognition systems to digital adversarial faces.

3. A new self-supervised framework, namely *FaceGuard*, for defending against adversarial face images. FaceGuard combines benefits of adversarial training, detection, and purification into a unified defense mechanism trained in an end-to-end manner.

4. A novel unified face attack detection framework, namely *UniFAD*, that automatically clusters similar attacks and employs a multi-task learning framework to jointly detect digital and

physical attacks. Proposed UniFAD allows for further identification of the attack categories, *i.e.*, whether attacks are adversarial, digitally manipulated, or contain spoof artifacts.

Chapter 2

Defending Against Face Spoofs

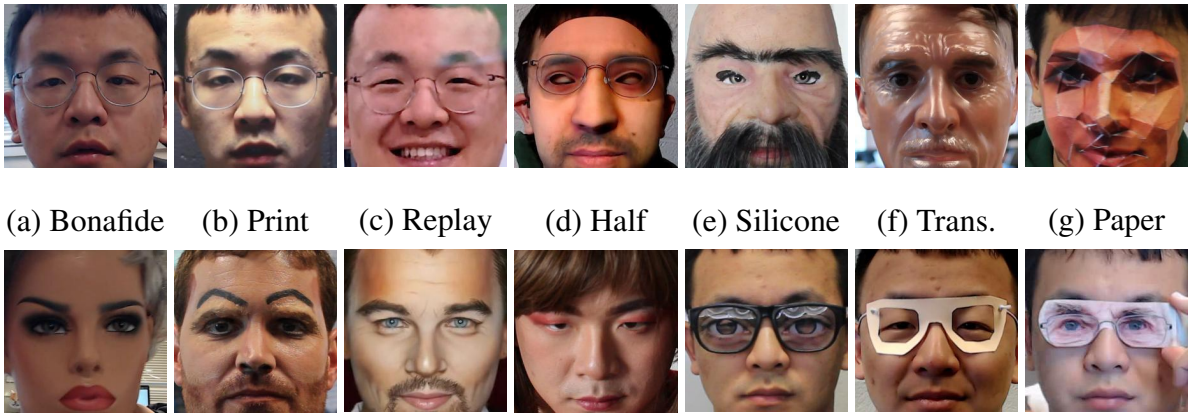
Face presentation attacks (physically crafted spoofs) challenge the robustness of face recognition systems. To safeguard AFR systems against spoofs, numerous presentation attack detection (PAD) methods have been proposed. In this chapter, we address two major issues with prevailing PAD approaches, namely, face PAD generalization and interpretability. Our main focus is to improve presentation attack detection performance across a wide variety of unknown attacks, while also maintaining high detection accuracy on spoofs encountered during the training of our proposed PAD solution.

2.1 Introduction

Despite impressive face recognition performance, current AFR systems remain vulnerable to the growing threat of *presentation attacks*[1]. Face spoofs are "fake faces" which can be constructed with a variety of different instruments (presentation attack instruments), *e.g.*, 3D printed masks, printed paper, or digital devices (video replay attacks from a mobile phone) with a goal of enabling an attacker to impersonate a victim's identity, or alternatively, obfuscate their own identity (see Figure 2.1). With the rapid proliferation of face images/videos on the Internet (especially on social

[1]ISO standard IEC 30107-1:2016(E) defines presentation attacks as *"presentation to the biometric data capture subsystem with the goal of interfering with the operation of the biometric system"* [64]. Note that these presentation attacks are different from digital manipulation of face images, such as DeepFakes [81] and adversarial faces [16].

media websites, such as Facebook, Twitter, or LinkedIn), replaying videos containing the victim's face or presenting a printed photograph of the victim to the AFR system is a trivial task [65]. Even if a face presentation attack detection system could trivially detect printed photographs and replay video attacks (*e.g.*, with depth sensors), attackers can still attempt to launch more sophisticated attacks such as 3D masks [66], make-up, or even virtual reality [67].



(a) Bonafide (b) Print (c) Replay (d) Half (e) Silicone (f) Trans. (g) Paper

(h) Mann. (i) Obf. (j) Imp. (k) Cosm. (l) Funny (m) PaperG (n) PaperC

Figure 2.1 Example presentation attack instruments: Simple attacks include (b) printed photograph, or (c) replaying the victim's video. More advanced presentation attacks can also be leveraged such as (d-h) 3D masks, (i-k) make-up attacks, or (l-n) partial attacks [30]. A bonafide face is shown in (a) for comparison. Here, the presentation attacks in (b-c, k-n) belong to the same person in (a).

The need for preventing face attacks is becoming increasingly urgent due to the user's privacy concerns associated with spoofed systems. Failure to detect face attacks can be a major security threat due to the widespread adoption of automated face recognition systems for border control [68]. In 2011, a young individual from Hong Kong boarded a flight to Canada disguised as an old man with a flat hat by wearing a silicone face mask to successfully fool the border control authorities [82].

Also consider that, with the advent of Apple's iPhone X and Samsung's Galaxy S8, all of us are carrying automated face recognition systems in our pockets embedded in our smartphones. Face recognition on our phones facilitates (i) unlocking the device, (ii) conducting financial transactions, and (iii) access to privileged content stored on the device. Failure to detect face presentation attacks

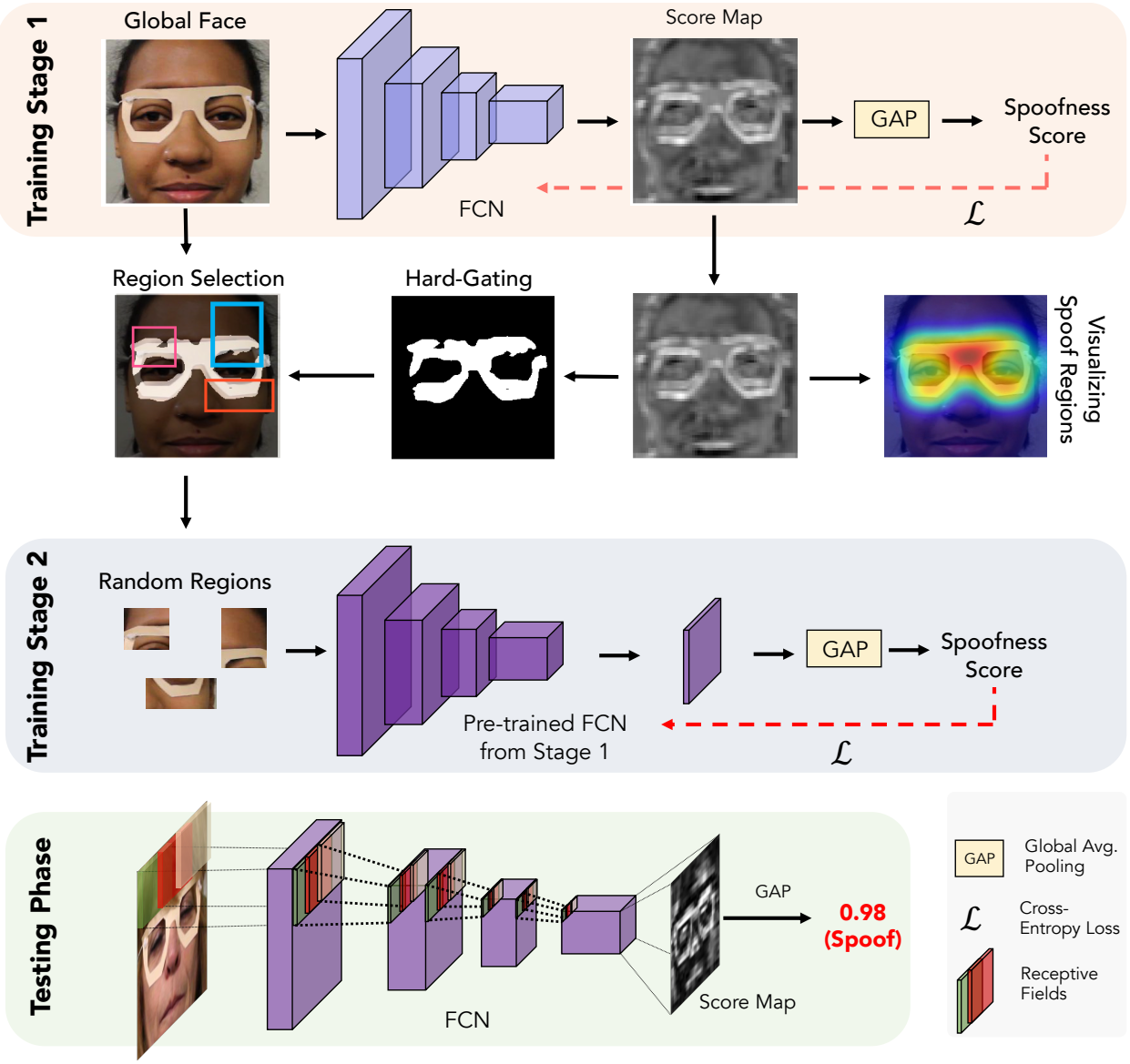


Figure 2.2 An overview of the proposed Self-Supervised Regional Fully Convolution Network (*SSR-FCN*). We train in two stages: (1) Stage 1 learns global discriminative cues via training on the entire face image. The score map obtained from stage 1 is hard-gated to obtain presentation attack regions in the face image. We randomly crop arbitrary-size patches from the presentation attack regions and fine-tune our network in stage 2 to learn local discriminative cues. During test, we input the entire face image to obtain the classification score. The score map can also be used to visualize the presentation attack regions in the input image.

on smartphones could compromise confidential information such as emails, banking records, social media content, and personal photos [83].

With numerous approaches proposed to detect face attacks, current face presentation attack

Table 2.1 A summary of publicly available face presentation attack detection datasets.

Dataset	Year	Statistics		# Presentation Attack Instruments				
		# Subj.	# Vids.	Replay	Print	3D Mask	Makeup	Partial
Attack [4]	2012	50	1,200	2	1	0	0	0
FASD [3]	2012	50	600	1	1	0	0	0
3DMAD [84]	2013	17	255	0	0	1	0	0
MFSD [85]	2015	35	440	2	1	0	0	0
Replay [86]	2016	40	1,030	1	1	0	0	0
MARs [87]	2016	35	1,009	0	0	2	0	0
Oulu-NPU [2]	2017	55	4,950	2	2	0	0	0
SiW [30]	2018	165	4,620	4	2	0	0	0
SiW-M [1]	2019	493	1,630	1	1	5	3	3

detection methods have following shortcomings:

Generalizabilty Since the exact presentation attack instrument may not be known beforehand, how to generalize well to "unknown"[2] attacks is of utmost importance. A majority of the prevailing state-of-the-art face presentation attack detection techniques focus only on detecting 2D printed paper and video replay attacks, and are vulnerable to presentation attacks crafted from materials not seen during training of the dectector. In fact, studies show a two-fold increase in error when presentation attack detection approaches encounter unknown presentation attack instruments [30]. In addition, current face presentation attack detection approaches rely on densely connected neural networks with a large number of learnable parameters (exceeding $2.7M$), where the lack of generalization across unknown presentation attack instruments is even more pronounced.

Lack of Interpretability Given a face image, face presentation attack detection approaches typically output a holistic face *"attack score"* which depicts the likelihood that the input image is bonafide or a presentation attack. Without an ability to visualize which regions of the face contribute to the overall decision made be the network, the global attack score alone may not be sufficient for a human operator to interpret the network's decision.

[2]Unseen attacks are presentation attack instruments that are known to the developers whereby algorithms can be specifically tailored to detect them, but their data is never used for training. Unknown attacks are presentation attack instruments that are not known to the developers and neither seen during training.

In an effort to impart generalizability and interpretability to face presentation attack detection systems, we propose a face presentation attack detection framework specifically designed to detect unknown presentation attack instruments, namely, **Self-Supervised Regional Fully Convolutional Network** (*SSR-FCN*). A Fully Convolutional Network (FCN) is first trained to learn global discriminative cues and automatically identify presentation attack regions in face images. The network is then fine-tuned to learn local representations via regional supervision. Once trained, the deployed model can automatically locate regions where attack occurs in the input image and provide a final attack score.

The contributions of this chapter are as follows:

- We show that features learned from local face regions have better generalization ability than those learned from the entire face image alone.
- We provide extensive experiments to show that the proposed approach, *SSR-FCN*, outperforms other local region extraction strategies and state-of-the-art face presentation attack detection methods on one of the largest publicly available dataset, namely, SiW-M, comprised of 13 different spresentation attack instruments. The proposed method reduces the Equal Error Rate (EER) by (i) 14% relative to state-of-the-art [88] under the unknown attack setting, and (ii) 40% on known presentation attack instruments. In addition, *SSR-FCN* achieves competitive performance on standard benchmarks on Oulu-NPU [2] dataset and outperforms prevailing methods on cross-dataset generalization (CASIA-FASD [3] and Replay-Attack [4]).
- The proposed *SSR-FCN* is also shown to be more interpretable since it can directly predict the parts of the faces that are considered as presentation attacks.

2.2 Background

In order to mitigate the threats associated with presentation attacks, numerous face presentation attack detection techniques, based on both software and hardware solutions, have been proposed.

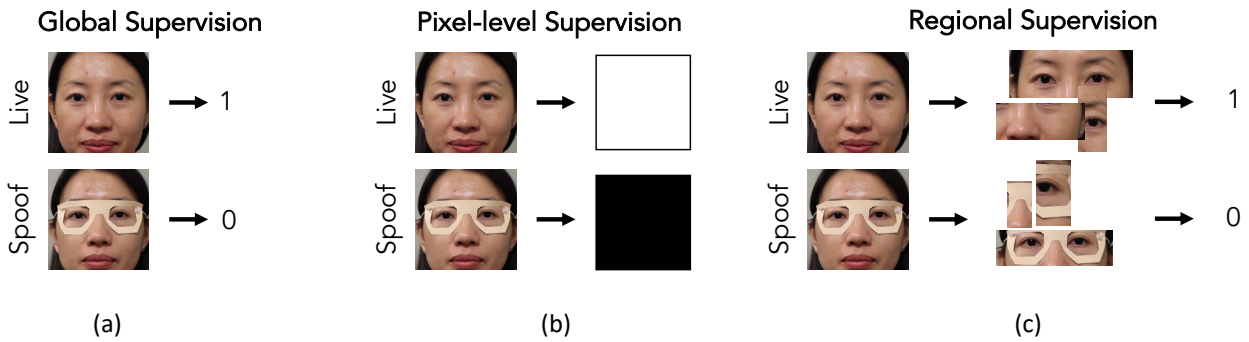


Figure 2.3 Illustration of drawbacks of prior approaches. Top: example of a bonafide face; Bottom: example of a paper glasses presentation attack. In this case, the presentation attack artifact is only present in the eye-region of the face. (a) Classifier trained with global supervision overfits to the bonafide class since both images are mostly bonafide (the presentation attack instrument covers only a part of the face). (b) Pixel-level supervision assumes the entire image is either bonafide or presentation attack and constructs label maps accordingly. This is not a valid assumption in mask, makeup, and partial presentation attack instruments. Instead, (c) the proposed framework, trains on extracted regions from face images. These regions can be based on domain knowledge, such as eye, nose, mouth regions, or randomly cropped. The proposed *SSR-FCN* utlizes self-supervised region-selection.

Early software-based solutions utilized liveness cues, such as eye blinking, lip movement, and head motion, to detect print attacks [89–92]. However, these approaches fail when they encounter unknown attacks such as printed attacks with cut eye regions (see Figure 2.1n). In addition, these methods require active cooperation of user in providing specific types of images making them tedious to use.

Since then, researchers have moved on to *passive* face presentation attack detection approaches that rely on texture analysis for distinguishing bonafide and presentation attacks, rather than motion or liveness cues. The majority of face presentation attack detection methods only focus on detecting print and replay attacks, which can be detected using features such as color and texture [93–98]. Many prior studies employ handcrafted features such as 2D Fourier Spectrum [85, 99], Local Binary Patterns (LBP) [94, 100–102], Histogram of oriented gradients (HOG) [93, 103], Difference-of-Gaussians (DoG) [104], Scale-invariant feature transform (SIFT) [96], and Speeded up robust features (SURF) [105]. Some techniques utilize presentation attack detection beyond the RGB color spectrum, such as incorporating the luminance and chrominance channels [102]. Instead of

a predetermined color spectrum, Li *et al.* [95] automatically learn a new color scheme that can best distinguish bonafide and presentation attacks. Another line of work extracts image quality features to detect presentation attacks [85, 97, 98]. Due to the assumption that presentation attack instruments are one of replay or print attacks, these methods severely suffer from generalization to unknown presentation attack instruments.

Hardware-based solutions in literature have incorporated 3D depth information [106–108], multi-spectral and infrared sensors [109], and even physiological sensors such as vein-flow information [110]. Presentation attack detection can be further enhanced by incorporating background audio signals [111]. However, with the inclusion of additional sensors along with a standard camera, the deployment costs can be exorbitant (*e.g.*, thermal sensors for iPhones cost over USD 400[3]).

State-of-the-art face presentation attack detection systems utilize Convolutional Neural Networks (CNN) so the feature set (representation) is learned that best differentiates bonafides from presentation attacks. Yang *et al.* were among the first to propose CNNs for face presentation attack detection and they showed about 70% decrease in Half Total Error Rate (HTER) compared to the baselines comprised of handcrafted features [112]. Further improvement in performance was achieved by directly modifying the network architecture [113–116]. Deep learning approaches also perform well for mask attack detection [66]. Incorporating auxiliary information (*e.g.* eye blinking) in deep networks can further improve the face presentation attack detection performance [30, 91].

Limited studies on generalizable face presentation attack detection focus on one-class classification approaches. These methods only model the distribution of bonafide face features using one-class classifiers such as one-class SVM [117], one-class GMM [118], or employ a distance metric loss [119]. However, these approaches have several drawbacks: (i) by only modeling the bonafide face feature distributions, the methods tend to overfit to bonafide class, and (ii) both one-class SVM and one-class GMM have been shown to perform poorly on public benchmark datasets (CASIA-FASD [3], Replay-Mobile [86], and MSU-MFSD [85]). A tree-based approach utilizing deep networks was proposed for generalizable face presentation attack detection [1]. In order to

[3]https://amzn.to/2zJ6YW4

prevent face presentation attack detection methods from overfitting to the specific subject, environment, and presentation attack instrument, transfer learning has also been studied [120–122]. However, these methods share similar network architecture that are densely connected with thirteen convolutional layers exceeding $2.7M$ learnable parameters [30, 88, 120–124]. Due to this, a majority of the aforementioned presentation attack detection methods also suffer from poor generalization performance.

Table 2.1 outlines the publicly available face presentation attack detection datasets.

2.3 Motivation

The approach is motivated by following observations:

2.3.1 Face Presentation Attack Detection is a Local Task

It is now generally accepted that for print and replay attacks, *"face presentation attack detection is usually a local task in which discriminative clues are ubiquitous and repetitive"* [125]. However, in the case of masks, makeups, and partial attacks, the ubiquity and repetitiveness of presentation attack cues may not hold true. For instance, in Figure 2.3 (a-c), presentation attack artifact (the paper glasses) are only present in the eye regions of the face. Unlike face recognition, face presentation attack detection does not require the entire face image in order to predict whether the image is a presentation attack or bonafide. In fact, our experimental results and their analysis will confirm that the entire face image alone can adverserly affect the convergence and generalization of networks.

2.3.2 Global vs. Local Supervision

Prior work can be partitioned into two groups: (i) *global supervision* where the input to the network is the entire face image and the CNN outputs a score indicating whether the image is bonafide or presentation attack [30, 88, 112–116, 123, 126], and (ii) *pixel-level supervision* where multiple

Table 2.2 Architecture details of the proposed FCN backbone.

Layer	# of Activations	# of Parameters
Input	$H \times W \times 3$	0
Conv	$H/2 \times W/2 \times 64$	$3 \times 3 \times 3 \times 64 + 64$
Conv	$H/4 \times W/4 \times 128$	$3 \times 3 \times 64 \times 128 + 128$
Conv	$H/8 \times W/8 \times 256$	$3 \times 3 \times 128 \times 256 + 256$
Conv	$H/16 \times H/16 \times 512$	$3 \times 3 \times 256 \times 512 + 512$
Conv	$H/16 \times H/16 \times 1$	$3 \times 3 \times 512 \times 1 + 1$
GAP	1	0
Total		**1.5M**

Conv and GAP refer to convolutional and global average pooling operations.

classification losses are aggregated over each pixel in the feature map [124, 127]. These studies assume that all pixels in the face image is either bonafide or presentation attack (see Figure 2.3(b)). This assumption holds true for presentation attack instruments, such as replay and print attacks (which are the only presentation attack instruments considered by the studies), but not for mask, makeup, and partial attacks. Therefore, pixel-level supervision can not only suffer from poor generalization across a diverse range of presentation attack instruments, but also convergence of the network is severely affected due to noisy labels.

In summary, based on the 13 different presentation attack instruments shown in Figure 2.1, for which we have the data, we gain the following insights: (i) face presentation attack detection is inherently a local task, and (ii) learning local representations can improve face presentation attack detection performance [124, 127]. Motivated by (i), we hypothesize that utilizing a Fully Convolutional Network (FCN) may be more appropriate for the face presentation attack detection task compared to a traditional CNN. The second insight suggests FCNs can be intrinsically regularized to learn local cues by enforcing the network to *look* at local spatial regions of the face. In order to ensure that these regions mostly comprise presentation attack patterns, we propose a *self-supervised* region extractor.

2.4 Proposed Approach

In this section, we describe the proposed *Self-Supervised Regional Fully Convolutional Network* (*SSR-FCN*) for generalized face presentation attack detection. As shown in Figure 2.2, we train the network in two stages, (a) Stage I learns global discriminative cues and predicts score maps, and (b) Stage II extracts arbitrary-size regions from presentation attack areas and fine-tunes the network via regional supervision.

2.4.1 Network Architecture

In typical image classification tasks, networks are designed such that information present in the input image can be used for learning *global* discriminative features in the form of a feature vector without utilizing the spatial arrangement in the input. To this end, a fully-connected (FC) layer is generally introduced at the end of the last convolutional layer. This fully-connected layer outputs a D-dimension feature that aggregates decisions at various spatial regions to obtain a global description of the input image. However, this is not ideal for partial presentation attacks since the presentation attack artifact is not present in all spatial regions. Given the plethora of available presentation attack instruments, it is better to learn *local* representations and make decisions on local spatial inputs rather than global descriptors. Therefore, we employ a Fully Convolutional Network (FCN) by replacing the FC layer in a traditional CNN with a 1×1 convolutional layer followed by a global average pooling layer. The FCN leads to three major advantages over traditional CNNs:

- **Arbitrary-sized inputs**: By replacing the fully-connected layer with a global average pooling layer, the entire FCN can accept input images of any image size. This property can be exploited to learn discriminative features at local spatial regions, regardless of the input size, rather than overfitting to a global representation of the entire face image.

- **Interpretability**: Since the proposed FCN is trained to provide decisions at a local level, the score map output by the network can be used to identify the presentation attack regions in the face.

Figure 2.4 Three presentation attack images and their corresponding binary masks extracted from predicted score maps. Black regions correspond to predicted bonafide regions, whereas, white regions indicate presentation attack.

- **Efficiency**: Via FCN, an entire face image can be inferred only once where local decisions are dynamically aggregated via the 1×1 convolution operator. Existing methods utilizing a traditional CNN which has a larger number of trainable parameters due to the fully connected layer at the end of the network. This necessitates a large training dataset which is limited in the face presentation attack detection literature (see Table 2.1). FCN is more parameter-efficient and can be trained in an effective manner (to avoid overfitting).

2.4.2 Network Efficiency

A majority of prior work on CNN-based face presentation attack detection employs architectures that are densely connected with thirteen convolutional layers [30, 88, 123, 124, 128]. Even with the placement of skip connections, the number of learnable parameters exceed $2.7M$. As we see in Table 2.1, only a limited amount of training data[4] is generally available in face presentation attack detection datasets. Limited data coupled with the large number of trainable parameters causes current approaches to overfit, leading to poor generalization performance under unknown attack scenarios. Instead, we employ a shallower neural network comprising of only five convolutional layers with approximately $1.5M$ learnable parameters (see Table 2.2).

2.4.3 Stage I: Training FCN Globally

We first train the FCN with global face images in order to learn global discriminative cues and identify presentation attack regions in the face image. Given an image, $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$, we detect a

[4]The lack of large-scale publicly available face presentation attack detection datasets is due to the time and effort required along with privacy concerns associated in acquiring such datasets.

face and crop the face region via 5 landmarks (two eyes, nose, and two mouth keypoints) in order to remove background information not pertinent to the task of face presentation attack detection. Here, H, W, and C refer to the height, width, and number of channels (3 in the case of RGB) of the input image. The face regions are then aligned and resized to a fixed-size (*e.g.*, 256×256) in order to maintain consistent spatial information across all training data.

The proposed FCN consists of four downsampling convolutional blocks each coupled with batch normalization and ReLU activation. The feature map from the fourth convolutional layer passes through a 1×1 convolutional layer. The output of the 1×1 convolutional layer represents a score map $\mathbf{S} \in \mathbb{R}^{(H_S \times W_S \times 1)}$ where each pixel in $\mathbf{S}$ represents a bonafide/presentation attack decision corresponding to its receptive field in the image. The height (H_S) and width (W_S) of the score map is determined by the input image size and the number of downsampling layers. For a $256 \times 256 \times 3$ input image, our proposed architecture outputs a $16 \times 16 \times 1$ score map.

The score map is then reduced to a single scalar value by globally average pooling. That is, the final classification score ($\mathbf{s}$) for an input image is obtained from the $(H_S \times W_S \times 1)$ score map ($\mathbf{S}$) by,

$$\mathbf{s} = \frac{1}{H_S \times W_S} \sum_{i}^{H_S} \sum_{j}^{W_S} \mathbf{S}_{i,j} \tag{2.4.1}$$

Using sigmoid activation on the final classification output ($\mathbf{s}$), we obtain a scalar $p(c|\mathbf{x}) \in [0, 1]$ predicting the likelihood that the input image is a presentation attack, where $c = 0$ indicates bonafide and $c = 1$ indicates presentation attack.

We train the network by minimizing the Binary Cross Entropy (BCE) loss,

$$\mathcal{L} = -\left[y \times log(p(c|\mathbf{x})) + (1 - y) \times log(1 - p(c|\mathbf{x}))\right] \tag{2.4.2}$$

where y is the ground truth label of the input image.

2.4.4 Stage II: Training FCN on Self-Supervised Regions

In order to supervise training at a local level, we propose a regional supervision strategy. We train the network to learn local cues by only showing certain regions of the face where presentation attack patterns exists. In order to ensure that presentation attack artifacts/patterns indeed exists within the selected regions, the pre-trained FCN from Stage I (2.4.3) can automatically guide the region selection process in presentation attack images. For bonafide faces, we can randomly crop a region from any part of the image.

Due to the absence of a fully connected layer, notice that FCN naturally encodes decisions at each pixel in feature map $\mathbf{S}$. In other words, higher intensity pixels within $\mathbf{S}$ indicate a larger likelihood of a presentation attack pattern residing within the receptive field in the image. Therefore, discriminative regions (presentation attack areas) are automatically highlighted in the score map by training on entire face images (see Figure 2.2).

We can then craft a binary mask M indicating the bonafide/presentation attack regions in the input presentation attack images. First, we soft-gate the score map by min-max normalization such that we can obtain a score map $\mathbf{S}' \in [0, 1]$.

Let $f_{\mathbf{S}'}(i, j)$ represent the activation in the (i, j)-th spatial location in the scaled score map $\mathbf{S}'$. The binary mask M is designed by hard-gating,

$$M(i,j) = \begin{cases} 1, & \text{if } \mathbf{S}'(i,j) \geq \tau \\ 0, & \text{otherwise} \end{cases} \tag{2.4.3}$$

where, τ is a threshold that controls the size of the hard-gated region ($\tau = 0.5$ in our case). A larger τ leads to smaller regions and smaller τ can lead to spurious presentation attack regions. Examples of binary masks are shown in Figure 2.4. From the binary mask, we can then randomly extract a rectangular bounding box such that the center of the rectangle lies within the detected presentation attack regions. In this manner, we can crop rectangular regions of arbitrary sizes from the input image such that each region contains presentation attack artifacts according to our

pre-trained global FCN. We constrain the width and height of the bounding boxes to be between MIN_{region} and MAX_{region}.

In this manner, we fine-tune our network to learn local discriminative cues.

2.4.5 Testing

Since FCNs can accept arbitrary input sizes and the fact that the proposed FCN has encountered entire faces in Stage I, we input the global face into the trained network and obtain the score map. The score map is then average pooled to extract the final classification output, which is then normalized by a sigmoid function in order to obtain an attack score within $[0, 1]$. That is, the final classification score is obtained by,

$$\frac{1}{1 + \exp(-\mathbf{s})}$$

In addition to the classification score, the score map ($\mathbf{S}$) can also be utilized for visualizing the presentation attack regions in the face by constructing a heatmap (see Figure 2.2).

2.5 Experimental Setup

2.5.1 Datasets

The following four datasets are utilized in our study (Table 2.1):

2.5.1.1 Spoof-in-the-Wild with Multiple Attacks (SiW-M) [1]

A dataset, collected in 2019, comprising 13 different presentation attack instruments, acquired specifically for evaluating generalization performance on unknown presentation attack instruments. Compared with other publicly available datasets (Table 2.1), SiW-M is diverse in presentation attack instruments, environmental conditions, and face poses. We evaluate *SSR-FCN* under both *unknown* and *known* settings, and perform ablation studies on this dataset.

2.5.1.2 Oulu-NPU [2]

A dataset comprised of 4,950 high-resolution video clips of 55 subjects. Oulu-NPU defines four protocols each designed for evaluating generalization against variations in capturing conditions, attack devices, capturing devices and their combinations. We use this dataset for comparing our approach with the prevailing state-of-the-art face presentation attack detection methods on the four protocols.

2.5.1.3 CASIA-FASD [3] & Replay-Attack [4]

Both datasets, collected in 2012, are frequently employed in face presentation attack detection literature for testing *cross-dataset generalization* performance. These two datasets provide a comprehensive collection of attacks, including warped photo attacks, cut photo attacks, and video replay attacks. Low-quality, normal-quality,and high-quality videos are recorded under different lighting conditions.

All images shown in this paper are from SiW-M testing sets.

2.5.2 Data Preprocessing

For all datasets, we first extract all frames in a video. The frames are then passed through MTCNN face detector [41] to detect 5 facial landmarks (two eyes, nose and two mouth corners). Similarity transformation is used to align the face images based on the five landmarks. After transformation, the images are cropped to 256×256. *All face images shown in the paper are cropped and aligned.* Before passing into network, we normalize the images by requiring each pixel to be within $[-1, 1]$ by subtracting 127.5 and dividing by 127.5.

2.5.3 Implementation Details

SSR-FCN is implemented in Tensorflow, and trained with a constant learning rate of $(1e-3)$ with a mini-batch size of 128. The objective function, $\mathcal{L}$, is minimized using Adam optimizer [129].

Table 2.3 Generalization error on learning global (CNN) vs. local (FCN) representations of SiW-M [1].

Method	Metric (%)	Replay	Obfuscation	Paper Glasses	Overall
CNN	ACER	13.3	**47.1**	32.2	30.8 ± 17.0
	EER	12.8	44.6	23.6	27.0 ± 13.2
FCN (Stage I)	ACER	**11.2**	52.2	**12.1**	$\mathbf{25.1 \pm 23.4}$
	EER	**11.2**	**37.6**	**12.4**	$\mathbf{20.4 \pm 12.1}$

It takes 20 epochs to converge. Following [30], we randomly initialize all the weights of the convolutional layers using a normal distribution of 0 mean and 0.02 standard deviation. We restrict the self-supervised regions to be at least $1/4$ of the entire image, that is, $MIN_{region} = 64$ and at most $MAX_{region} = 256$ which is the size of the global face image. Data augmentation during training involves random horizontal flips with a probability of 0.5. We train and test our proposed method on a single Nvidia GTX 1080Ti GPU. For evaluation, we compute the attack scores for all frames in a video and temporally average them to obtain the final classification score.

2.5.4 Evaluation Metrics

For all the experiments, we report the standard ISO/IEC 30107 [64] metrics:

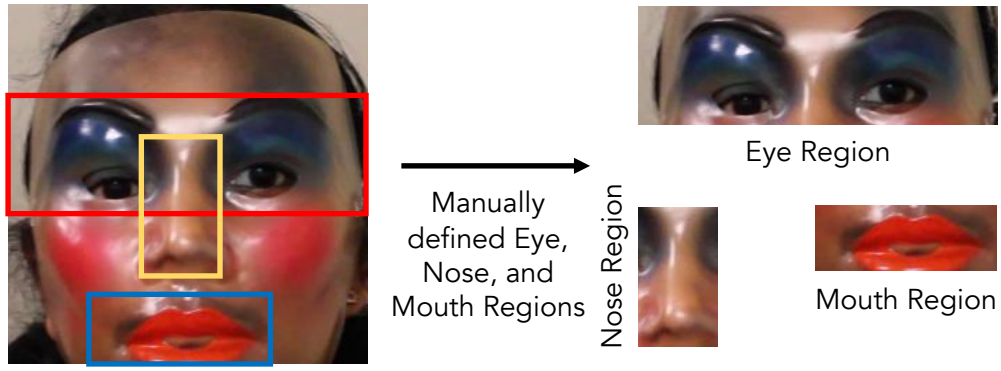
1. Attack Presentation Classification Error Rate (APCER): the worst error rate among all the presentation attack instruments;

2. Bonafide Presentation Classification Error Rate (BPCER):

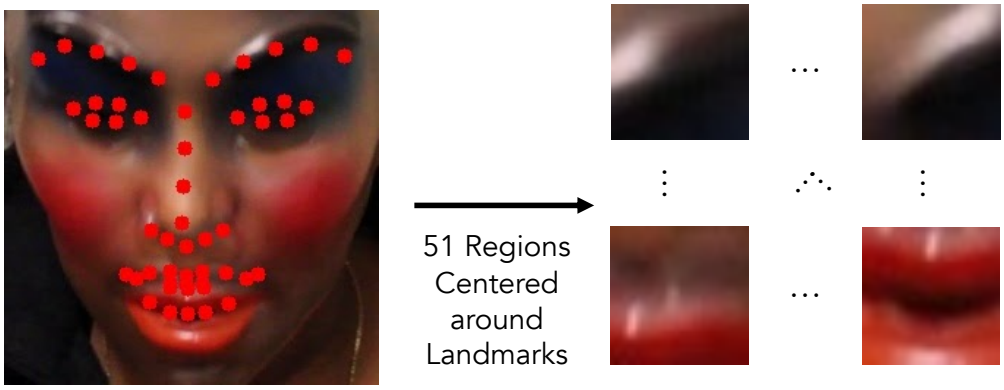
3. Average Classification Error Rate (ACER): the mean of APCER and BPCER.

In addition, we also report the Equal Error Rate (EER) and the True Detection Rate (TDR) at 2.0%[5] False Detection Rate (FDR) for our evaluation. For a fair comparison with prior work, we report the Half Total Error Rate (HTER) for cross-dataset evaluation. *Except for EER and HTER, we employ a decision threshold of* 0.5.

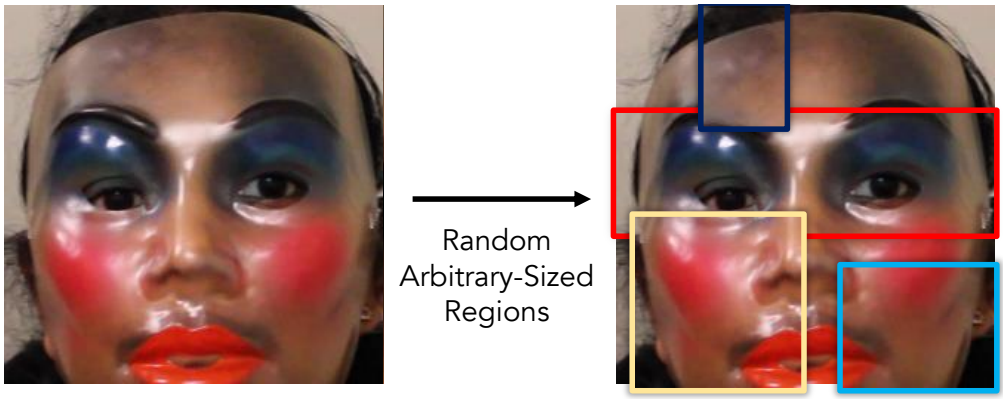
[5]Due to the small number of bonafide samples, thresholds at lower False Detection Rate (FDR) such as 0.2% (recommended under the IARPA ODIN program) cannot be computed.



(a) Manually-defined Regions



(b) Landmark Based



(c) Self-Supervised (Proposed)

Figure 2.5 Illustration of various region extraction strategies from training images. (a) and (b) are regions extracted via domain knowledge (manually defined) or landmark-based. (c) random regions extracted via proposed self-supervision scheme. Each color denotes a separate region.

Table 2.4 Generalization performance of different region extraction strategies on SiW-M dataset. Here, each column represents an unknown presentation attack instrument while the method is trained on the remaining 12 presentation attack instruments.

		Replay	Print	Mask Attacks					Makeup Attacks			Partial Attacks			
Method	**Metric** (%)	 **Replay** 99 vids.	 **Print** 118 vids.	 **Half** 72 vids.	 **Silicone** 27 vids.	 **Trans.** 88 vids.	 **Paper** 17 vids.	 **Mann.** 40 vids.	 **Obf.** 23 vids.	 **Imp.** 61 vids.	 **Cosm.** 50 vids.	 **FunnyEye** 160 vids.	 **Glasses** 127 vids.	 **Paper Cut** 86 vids.	**Mean ± Std.**
Global	ACER	11.2	15.5	12.8	21.5	35.4	6.1	10.7	52.2	50.0	20.5	26.2	**12.1**	9.6	22.6 ± 15.3
	EER	11.2	14.0	12.8	23.1	26.6	2.9	11.0	37.6	10.4	17.0	24.2	12.4	10.1	16.8 ± 9.3
Eye-Region	ACER	13.2	13.7	7.5	17.4	**22.5**	5.79	6.2	19.5	8.3	11.7	32.8	15.3	7.3	13.2 ± 8.5
	EER	12.4	11.4	7.3	15.2	21.5	2.9	6.5	20.2	7.8	11.2	27.2	14.7	7.5	12.3 ± 6.2
Nose-Region	ACER	17.4	10.5	8.2	13.8	30.3	5.3	8.4	37.4	5.1	18.0	35.5	31.4	7.1	17.6 ± 12.0
	EER	14.6	9.8	9.2	12.7	22.0	5.2	8.4	23.6	4.4	14.6	24.9	27.7	7.6	14.2 ± 7.9
Mouth-Region	ACER	20.5	20.7	22.9	26.3	30.6	15.6	17.1	44.2	18.1	24.0	38.0	47.2	8.5	25.7 ± 11.4
	EER	19.9	21.3	22.6	25.1	30.0	10.1	10.7	40.9	16.1	24.0	35.5	40.4	8.1	23.4 ± 10.9
Global + Eye + Nose	ACER	10.9	10.5	7.5	17.7	28.7	**5.1**	7.0	38.0	5.1	13.6	29.4	15.2	6.2	15 ± 10.7
	EER	10.2	10.0	7.7	15.8	21.3	1.8	6.7	21.0	**3.0**	12.3	22.5	**12.3**	6.5	11.6 ± 6.8
Landmark Region	ACER	10.7	9.2	18.4	25.1	26.4	6.2	6.9	53.8	8.1	15.4	35.8	40.8	7.6	20.3 ± 15.2
	EER	8.0	10.1	12.2	23.1	18.8	8.9	4.1	40.1	9.9	15.6	**17.7**	25.6	4.9	15.3 ± 10
Global + Landmark	ACER	12.0	11.2	7.3	23.7	26.4	6.3	5.9	26.7	6.7	**10.7**	27.8	25.7	6.4	15.1 ± 9.2
	EER	11.5	10.1	7.2	19.0	**4.9**	6.6	4.6	25.6	6.7	**10.9**	23.5	18.5	4.7	11.8 ± 7.4
Random-Crop	ACER	9.2	**6.7**	7.3	19.9	30.9	9.1	6.9	44.0	6.5	13.8	31.8	28.6	**5.9**	17.0 ± 12.8
	EER	8.9	**7.8**	10.3	17.9	21.3	3.7	6.5	32.7	5.4	13.7	18.7	19.4	7.1	13.3 ± 8.3
Global + Random-Crop	ACER	12.3	10.7	6.5	18.2	22.9	6.2	6.1	**18.6**	**4.9**	11.6	32.7	16.1	7.2	13.4 ± 8.1
	EER	10.9	9.2	6.9	16.6	21.3	2.9	5.2	18.8	3.7	11.5	19.0	14.9	6.2	11.3 ± 6.3
SSR-FCN	ACER	**7.4**	19.5	**3.2**	**7.7**	33.3	5.2	**3.3**	22.5	5.9	11.7	**21.7**	14.1	6.4	**12.4 ± 9.2**
	EER	**6.8**	11.2	**2.8**	**6.3**	28.5	**0.4**	**3.3**	**17.8**	3.9	11.7	21.6	13.5	**3.6**	**10.1 ± 8.4**

2.6 Experimental Results

2.6.1 Evaluation of Global Descriptor vs. Local Representation

In order to analyze the impact of learning local embeddings as opposed to learning a global embedding, we conduct an ablation study on three presentation attack instruments in the SiW-M dataset [1], namely, Replay (Figure 2.1c), Obfuscation (Figure 2.1i), and Paper Glasses (Figure 2.1m). *The experiment is conducted under the unknown attack scenario (leave-one-instrument-out protocol).*

In this experiment, a *traditional CNN* learning a global image descriptor is constructed by replacing the 1×1 convolutional layer with a fully connected layer. We compare the CNN to the proposed backbone *FCN* in Table 2.2 which learns local representations. For a fair comparison between CNN and FCN, we utilize the same meta-parameters and employ global supervision only (Stage I).

In Table 2.3, we find that overall FCNs are more generalizable to unknown presentation attack instruments compared to global embeddings. For presentation attack instruments where the presentation attack affects the entire face, such as replay attacks, the differences between generalization performance of CNN and FCN are negligible. Here, presentation attack decisions at local spatial regions do not have any significant advantage over a single presentation attack decision over the entire image. Recall that CNNs employ a fully connected layer which strips away all spatial information. This explains why local decisions can significantly improve generalizability of FCN over CNN when presentation attack instruments are local in nature (*e.g.*, make-up attacks and partial attacks). Due to subtlety of obfuscation attacks and localized nature of paper glasses, FCN can exhibit a relative reduction in EER by 16% and 47%, respectively, relative to CNN.

2.6.2 Region Extraction Strategies

We considered 6 different region extraction strategies, namely, *Eye-Region*, *Nose-Region*, *Mouth-Region*, *Landmark-Region*, *Random-Crop*, and *Self-Supervised Regions (Proposed)* (see Figure 2.5). We also include results for a *Global* model which refers to training the FCN with the entire face image only (Stage I).

Since all face images are aligned and cropped, spatial information is consistent across all images in datasets. Therefore, we can automatically extract facial regions that inclue eye, nose, and mouth regions (Figure 2.5a). We train the proposed FCN separately on each of the three regions to obtain three models: eye-region, nose-region, and mouth-region.

We also investigate extracting regions defined by face landmarks. For this, we utilize a state-of-the-art landmark extractor, namely DLIB [130], to obtain 68 landmark points. We exclude 17 landmarks defined around the jawline and define a subset of 51 landmark points around eyebrows (10 landmarks), eyes (12 landmarks), nose (9 landmarks), and mouth (20 landmarks). A total of 51 regions (with a fixed size 32×32) centered around each landmark are extracted and used train a single FCN on all 51 regions.

Our findings are as follows: (i) almost all methods with regional supervision have lower overall

error rates as compared to training with the entire face. Exception to this is when FCN is trained only on mouth regions. This is likely because a majority of presentation attack instruments may not contain presentation attack patterns across mouth regions (Figure 2.1). (ii) when both global and domain-knowledge strategies (specifically, eyes and nose) are fused, the generalization performance improves compared to the global model alone. Note that we do not fuse the mouth region since the performance is poor for mouth regions. Similarly, we find that regions cropped around landmarks when fused with the global classifier can achieve better generalization performance. (iii) sampling random local regions of the face also results in high error rates across the diverse set of presentation attack instruments. (iv) compared to all region extraction strategies, the proposed self-supervised region extraction strategy (Stage I $\rightarrow$ Stage II) achieves the lowest generalization error rates across all presentation attack instruments with a 40% and 45% relative reduction in EER and ACER compared to the Global model (Stage I). This supports our hypothesis that both Stage I and Stage II are required for enhanced generalization performance across unknown presentation attack instruments. A score-level fusion of the global FCN with self-supervised regions does not show any significant reduction in error rates. This is because we already trained the proposed FCN on global faces in Stage I.

The *Random-Crop* strategy can be viewed as a variant of training the proposed Stage II without any region-selection guidance from Stage I. In order to further investigate the benefit of the proposed self-supervision method, we train two models to distinguish between bonafide samples and *Paper Glasses* presentation attack samples: (i) *Random Crop* model is trained on patches randomly sampled from the input image such that the size of each region is between 64×64 and 256×256, and (ii) *Self-Supervised Regions* model is trained on patches sampled from weakly labeled regions by the pre-trained model from Stage I. For a fair comparison, we do not employ the pre-trained model from Stage I to train the *Self-Supervised Regions* model. In Figure 2.7, we plot the training loss for both models over multiple training iterations. We can observe that the proposed self-supervised region method aids in network convergence. This is because random cropping can result in training with bonafide regions from presentation attack samples (see Figure 2.7), whereas, the proposed

(a) Obfuscation (b) Attack Score: 1.0 (c) Attack Score: 0.7

(d) Obfuscation (e) Attack Score: 0.0 (f) Attack Score: 0.0

Figure 2.6 (a) An example obfuscation presentation attack attempt where our network correctly predicts the input to be a presentation attack. (b, e) Score map output by our network trained via Self-Supervised Regions. (c, f) Score map output by FCN trained on entire face images. (d) An example obfuscation presentation attack attempt where our network *incorrectly* predicts the input to be a bonafide. Attack scores are given below the score maps. Decision threshold is 0.5.

self-supervision method ensures that regions sampled from presentation attacks indeed contains presentation attack artifacts.

In Figure 2.6, we analyze the effect of training the FCN locally vs. globally on the prediction results. In the first row, where both models correctly predict the input to be a presentation attack, we see that FCN trained via random regions can correctly identify presentation attack regions such as fake eyebrows. In contrast, the global FCN can barely locate the fake eyebrows. Since random regions increases the variability in the training set along with advantage of learning local features than global FCN, we find that proposed self-supervised regional supervision performs best.

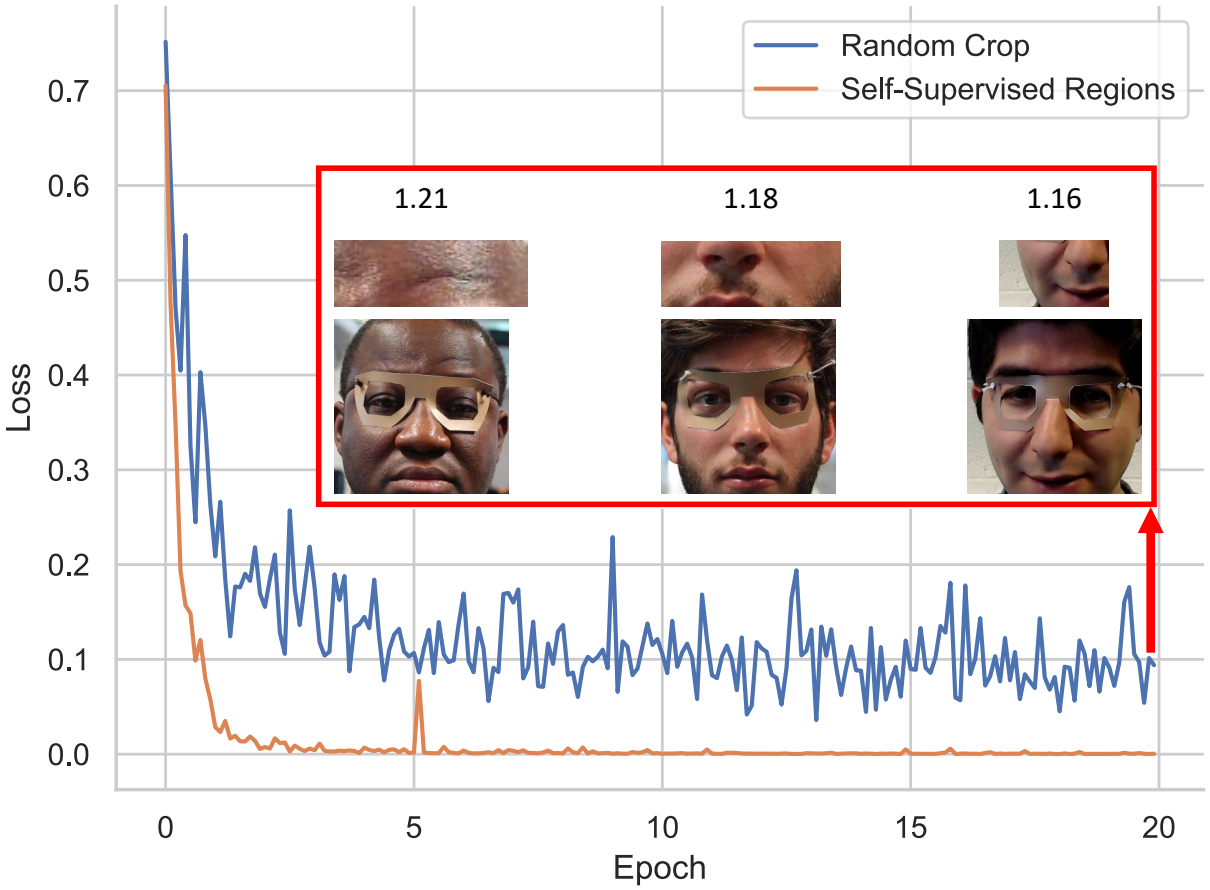


Figure 2.7 Network convergence over a number of training iterations when a model trains on (a) randomly cropped patches (blue line), and (b) self-supervised regions extracted via pre-trained model from Stage I (orange line). Randomly cropping patches may result in noisy samples where bonafide samples from presentation attack samples may be used for training. Some example randomly cropped patches with high training loss are shown above the lines. Instead, we find that the proposed self-supervision aids in network convergence.

2.6.3 Evaluation of Network Capacity

In Table 2.5, we show the generalization performance of our model when we vary the capacity of the network. We consider three different variants of the proposed FCN: (a) 3-layer FCN ($76K$ parameters), (b) 5-layer FCN ($1.5M$ parameters; proposed), and (c) 6-layer FCN ($3M$ parameters). This experiment is evaluated on three unknown presentation attack instruments, namely, *Replay*, *Obfuscation*, and *Paper Glasses*. We chose these presentation attack instruments due to their vastly diverse nature. Replay attacks consist of global presentation attack patterns, whereas

Table 2.5 Generalization error of FCNs with respect to the number of trainable parameters.

Method	Metric (%)	Replay	Obfuscation	Paper Glasses	Mean ± Std.
3-layers ($76K$)	ACER	13.9	57.6	12.3	27.9 ± 25.7
	EER	14.0	44.1	7.5	21.9 ± 19.5
5-layers ($1.5M$; proposed)	ACER	**7.4**	**22.5**	**14.1**	**14.7 ± 7.6**
	EER	**6.8**	**17.8**	**13.5**	**12.7 ± 4.5**
6-layers ($3M$)	ACER	11.2	32.9	19.8	21.3 ± 11.0
	EER	7.8	25.19	19.7	17.6 ± 7.2

Table 2.6 Results on SiW-M: Unknown Attacks. Here, each column represents an unknown presentation attack instrument while the method is trained on the remaining 12 presentation attack instruments.

		Replay	Print	Mask Attacks					Makeup Attacks			Partial Attacks			
															
Method	**Metric**	**Replay** 99 vids.	**Print** 118 vids.	**Half** 72 vids.	**Silicone** 27 vids.	**Trans.** 88 vids.	**Paper** 17 vids.	**Mann.** 40 vids.	**Obf.** 23 vids.	**Imp.** 61 vids.	**Cosm.** 50 vids.	**FunnyEye** 160 vids.	**Glasses** 127 vids.	**Paper Cut** 86 vids.	**Mean ± Std.**
SVM+LBP [2]	ACER	20.6	18.4	31.3	21.4	45.5	11.6	13.8	59.3	23.9	16.7	35.9	39.2	11.7	26.9 ± 14.5
	EER	20.8	18.6	36.3	21.4	37.2	7.5	14.1	51.2	19.8	16.1	34.4	33.0	7.9	24.5 ± 12.9
Auxiliary [30]	ACER	16.8	6.9	19.3	14.9	52.1	8.0	12.8	55.8	13.7	11.7	49.0	40.5	5.3	23.6 ± 18.5
	EER	14.0	4.3	11.6	12.4	24.6	7.8	10.0	72.3	10.1	**9.4**	21.4	18.6	4.0	17.0 ± 17.7
DTN [1]	ACER	9.8	**6.0**	15.0	18.7	36.0	4.5	7.7	48.1	11.4	14.2	**19.3**	19.8	8.5	16.8 ± 11.1
	EER	10.0	**2.1**	14.4	18.6	26.5	5.7	9.6	50.2	10.1	13.2	**19.8**	20.5	8.8	16.1 ± 12.2
CDC [88]	ACER	10.8	7.3	9.1	10.3	**18.8**	**3.5**	5.6	42.1	**0.8**	14.0	24.0	17.6	**1.9**	12.7 ± 11.2
	EER	9.2	5.6	4.2	11.1	**19.3**	5.9	5.0	43.5	**0.0**	14.0	23.3	14.3	**0.0**	11.9 ± 11.8
Proposed	ACER	**7.4**	19.5	**3.2**	**7.7**	33.3	5.2	**3.3**	**22.5**	5.9	**11.7**	21.7	**14.1**	6.4	**12.4 ± 9.2**
	EER	**6.8**	11.2	**2.8**	**6.3**	28.5	**0.4**	**3.3**	**17.8**	3.9	11.7	21.6	**13.5**	3.6	**10.1 ± 8.4**
	TDR*	72.0	51.0	96.0	55.9	39.0	100.0	95.0	31.0	90.0	44.0	33.0	42.9	94.7	**65.0 ± 25.9**

*TDR evaluated at 2.0% FDR

Table 2.7 Results on SiW-M: Known presentation attack instruments.

Method	Metric (%)			Mask Attacks					Makeup Attacks			Partial Attacks			Mean ± Std.
		Replay	Print	Half	Silicone	Trans.	Paper	Mann.	Obf.	Imp.	Cosm.	Funny Eye	Glasses	Paper Cut	
Auxiliary [30]	ACER	5.1	5.0	5.0	10.2	5.0	9.8	6.3	19.6	5.0	26.5	5.5	5.2	5.0	8.7 ± 6.8
	EER	4.7	**0.0**	1.6	10.5	4.6	10.0	6.4	12.7	**0.0**	19.6	7.2	7.5	0.0	6.5 ± 5.8
Proposed	ACER	**3.5**	**3.1**	**1.9**	**5.7**	**2.1**	**1.9**	**4.2**	**7.2**	**2.5**	**22.5**	**1.9**	**2.2**	**1.9**	**4.7 ± 5.6**
	EER	**3.5**	3.1	**0.1**	**9.9**	**1.4**	**0.0**	**4.3**	**6.4**	2.0	**15.4**	**0.5**	**1.6**	**1.7**	**3.9 ± 4.4**
	TDR*	55.5	92.3	69.5	100.0	90.4	100.0	85.1	92.5	78.7	99.1	95.6	95.7	76.0	87.0 ± 13.0

*TDR evaluated at 2.0% FDR

obfuscation attacks are extremely subtle cosmetic changes. Paper Glasses are constrained only to eyes. While a large number of trainable parameters lead to poor generalization due to over-

fitting to the presentation attack instruments seen during training, whereas, too few parameters limits learning discriminative features. Based on this observation and experimental results, we utilize the 5-layer FCN (see Table 2.2) with approximately 1.5M parameters. A majority of prior studies employ 13 densely connected convolutional layers with trainable parameters exceeding $2.7M$ [30, 88, 124, 127].

2.6.4 Generalization across Unknown Attacks

The primary objective of this work is to enhance generalization performance across a multitude of unknown presentation attack instruments in order to effectively gauge the expected error rates in real-world scenarios. The evaluation protocol in SiW-M follows a leave-one-spoof-out testing protocol where the training split contains 12 different presentation attack instruments and the 13^{th} presentation attack instrument is held out for testing. Among the bonafide videos, 80% are kept in the training set and the remaining 20% is used for testing bonafide. Note that there are no overlapping subjects between the training and testing sets. Also note that no data sample from the testing presentation attack instrument is used for validation since we evaluate our approach under unknown attacks. We report ACER and EER across the 13 splits. In addition to ACER and EER, we also report the TDR at 2.0% FDR.

In Table 2.6, we compare *SSR-FCN* with prior work. We find that our proposed method achieves significant improvement in comparison to the published results [88] (relative reduction of 14% on the average EER and 3% on the average ACER). Note that the standard deviation across all 13 presentation attack instruments is also reduced compared to prior approaches, even though some of them [30, 88] utilize auxiliary data such as depth and temporal information.

Specifically, we reduce the EERs of replay, half mask, transparent mask, silicone mask, paper mask, mannequin head, obfuscation, impersonation, and paper glasses relatively by 27%, 33%, 43%, 93%, 34%, 59%, and 6%, respectively. Among all the 13 presentation attack instruments, detecting obfuscation attacks is the most challenging. This is due to the fact that the makeup applied in these attacks are very subtle and majority of the faces are bonafide. Prior works were

not successful in detecting these attacks and predict most of the obfuscation attacks as bonafide. By learning discriminative features locally, our proposed network improves the state-of-the-art obfuscation attack detection performance by 59% in terms of EER and 46% in terms of ACER.

2.6.5 SiW-M: Detecting Known Attacks

Here all the 13 presentation attack instruments in SiW-M are used for training as well as testing. We randomly split the SiW-M dataset into a 60%-40% training/testing split and report the results in Table 2.7. In comparison to a state-of-the-art face presentation attack detection method [30], our method outperforms for almost all of the individual presentation attack instruments as well as the overall performance across presentation attack instruments. *Auxiliary [30]* utilizes depth and temporal information for presentation attack detection which adds significant complexity to the network. We find that characterizing local spatial regions as bonafide/presentation attack in fact leads to better generalization on unknown attacks (see Table 2.6) and specialization on known attacks.

2.6.6 Evaluation on Oulu-NPU Dataset

We follow the four standard protocols defined in the OULU-NPU dataset [2] which cover the cross-background, cross-presentation-attack-instrument (cross-PAI), cross-capture-device, and cross-conditions evaluations:

- **Protocol I:** unseen subjects, illumination, and backgrounds;
- **Protocol II:** unseen subjects and attack devices;
- **Protocol III:** unseen subjects and cameras;
- **Protocol IV:** unseen subjects, illumination, backgrounds, attack devices, and cameras.

We compare the proposed *SSR-FCN* with the best performing method, namely GRADIENT [131], in IJCB Mobile Face Anti-Spoofing Competition [131] for each protocol. We

Table 2.8 Error Rates (%) of the proposed *SSR-FCN* and and competing face presentation attack detectors under the four standard protocols of Oulu-NPU [2].

Protocol	Method	APCER	BPCER	ACER
I	GRADIENT [131]	1.3	12.5	6.9
	Auxiliary [30]	1.6	1.6	1.6
	DeepPixBiS [124]	**0.8**	**0.0**	**0.4**
	TSCNN-ResNet [132]	5.1	6.7	5.9
	SSR-FCN (Proposed)	1.5	7.7	4.6
II	GRADIENT [131]	3.1	1.9	**2.5**
	Auxiliary [30]	**2.7**	2.7	2.7
	DeepPixBiS [124]	11.4	**0.6**	6.0
	TSCNN-ResNet [132]	7.6	2.2	4.9
	SSR-FCN (Proposed)	3.1	3.7	3.4
III	GRADIENT [131]	**2.1 ± 3.9**	5.0 ± 5.3	3.8 ± 2.4
	Auxiliary [30]	2.7 ± 1.3	3.1 ± 1.7	2.9 ± 1.5
	DeepPixBiS [124]	11.7 ± 19.6	10.6 ± 14.1	11.1 ± 9.4
	TSCNN-ResNet [132]	3.9 ± 2.8	7.3 ± 1.1	5.6 ± 1.6
	SSR-FCN (Proposed)	2.9 ± 2.1	**2.7 ± 3.2**	**2.8 ± 2.2**
IV	GRADIENT [131]	**5.0 ± 4.5**	15.0 ± 7.1	10.0 ± 5.0
	Auxiliary [30]	9.3 ± 5.6	10.4 ± 6.0	**9.5 ± 6.0**
	DeepPixBiS [124]	36.7 ± 29.7	13.3 ± 16.8	25.0 ± 12.7
	TSCNN-ResNet [132]	11.3 ± 3.9	**9.7 ± 4.8**	9.8 ± 4.2
	SSR-FCN (Proposed)	8.3 ± 6.8	13.3 ± 8.7	10.8 ± 5.1

also include some newer baseline methods, including Auxiliary [30], DeepPixBiS [124], and TSCNN [132]. We compare our proposed method with 10 baselines in total for each protocol. Additional baselines can be found in supplementary material.

In Table 2.8, *SSR-FCN* achieves ACERs of 4.6%, 3.4%, 2.8%, and 10.8% in the four protocols, respectively. Among the baselines, *SSR-FCN* even outperforms prevailing state-of-the-art methods

Table 2.9 Cross-Dataset HTER (%) of the proposed *SSR-FCN* and competing face presentation attack detectors.

Method	**CASIA → Replay**	**Replay → CASIA**
CNN [112]	48.5	45.5
Color Texture [102]	47.0	49.6
FaceSpoofBuster [133]	43.3	53.0
Auxiliary [30]	27.6	**28.4**
De-Noising [125]	28.5	41.1
Damer & Dimitrov [134]	28.4	38.1
STASN [135]	31.5	30.9
SAPLC [127]	27.3	37.5
SSR-FCN (Proposed)	**19.9**	41.9

"CASIA → Replay" denotes training on CASIA and testing on Replay-Attack

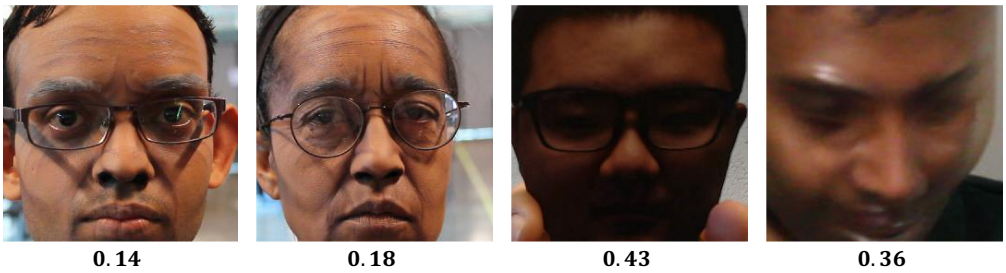
in protocol III which corresponds to generalization performance for unseen subjects and cameras. The results are comparable to baseline methods in the other three protocols. Since Oulu-NPU comprises of only print and replay attacks, a majority of the baseline methods incorporate auxiliary information such as depth and motion. Indeed, incorporating auxiliary information could improve the results at the risk of overfitting and overhead cost and time.

2.6.7 Cross-Dataset Generalization

In order to evaluate the generalization performance of *SSR-FCN* when trained on one dataset and tested on another, following prior studies, we perform a cross-dataset experiment between CASIA-FASD [3] and Replay-Attack [4].

In Table 2.9, we find that, compared to 6 prevailing state-of-the-art methods, the proposed *SSR-FCN* achieves the lowest error (a 27% improvement in HTER) when trained on CASIA-FASD [3] and evaluated on Replay-Attack [4]. On the other hand, *SSR-FCN* achieves worse performance when trained on Replay-Attack and tested on CASIA-FASD. This can likely be attributed to higher resolution images in CASIA-FASD compared to Replay-Attack. This demonstrates that *SSR-FCN* trained with higher-resolution data can generalize better on poorer quality testing images,

(a) Misclassified Lives as Spoofs



(b) Misclassified Spoofs as Lives

Figure 2.8 Example cases where the proposed framework, *SSR-FCN*, fails to correctly classify bonafides and presentation attacks. (a) Bonafides are misclassified as presentation attacks likely due to bright lighting and occlusions in face regions. (b) Presentation attacks misclassified as bonafides due to the subtle nature of make-up attacks and transparent masks. Corresponding attack scores ($\in [0, 1]$) are provided below each image. Larger value of attack score indicates a higher likelihood that the input image is a presentation attack. Decision threshold is 0.5.

but the reverse may not hold true. We intend on addressing this limitation in future work.

Additional baselines can be found in supplementary material.

2.6.8 Failure Cases

Even though experiment results show enhanced generalization performance, our model still fails to correctly classify certain input images. In Figure 2.8, we show a few such examples.

Figure 2.8a shows incorrect prediction of bonafides as presentation attacks in the presence of inhomogeneous illumination. This is because the model predicts bonafide as being one of replay and print attacks which exhibit bright lighting patterns due to the recapturing media such as smartphones and laptops. Since we fine-tune our network via regional supervision in Stage II, artifacts that obstruct parts of the faces can also adversely affect our model.

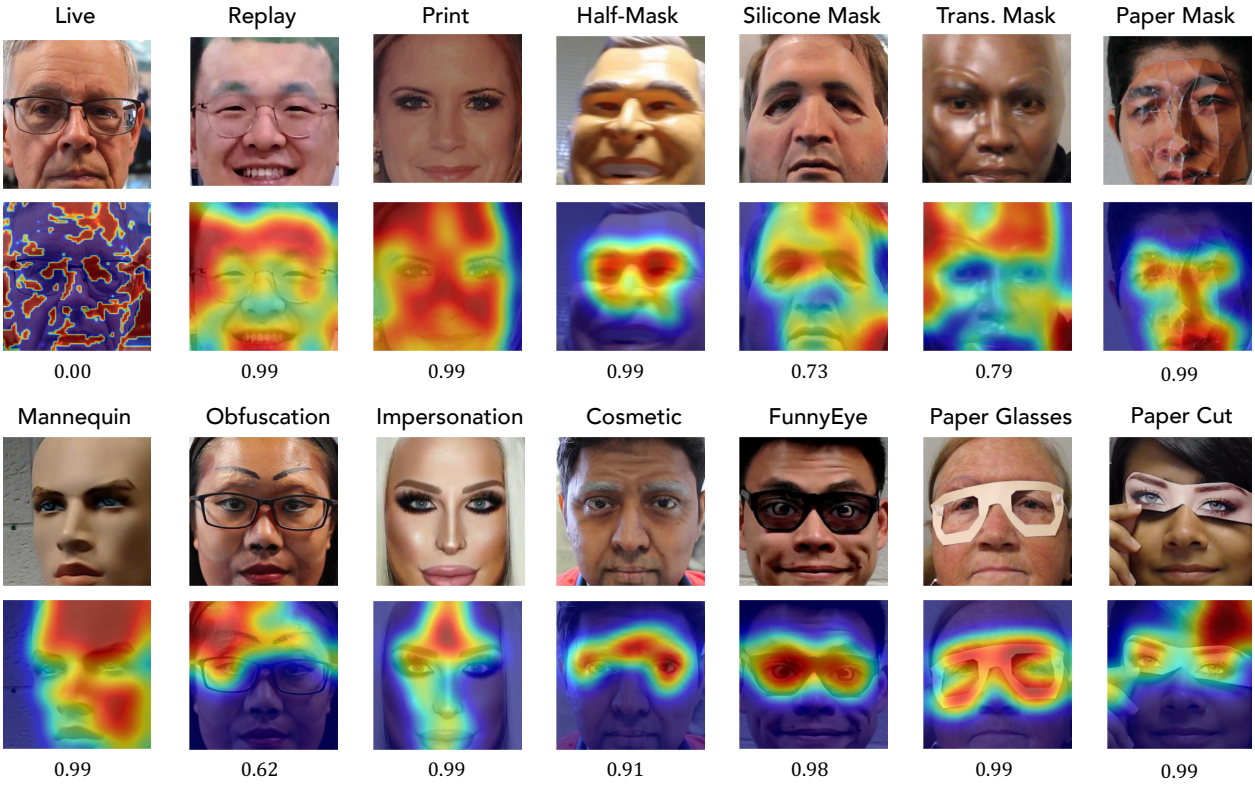


Figure 2.9 Visualizing presentation attack regions via the proposed *SSR-FCN*. Red regions indicate higher likelihood of being a presentation attack region. Corresponding attack scores ($\in [0, 1]$) are provided below each image. Larger value of attack score indicates a higher likelihood that the input image is a presentation attack. Decision threshold is 0.5.

Figure 2.8b shows incorrect classification of presentation attack as bonafides. This is particularly true when presentation attack artifacts are very subtle, such as cosmetic and obfuscation make-up attacks. Transparent masks can also be problematic when the mask itself is barely visible.

2.6.9 Computational Requirement

Since face presentation attack detection modules are employed as a pre-processing step for automated face recognition systems, it is crucial that the presentation attack prediction time should be as low as possible. The proposed approach comprises of 1.5M trainable parameters compared to a traditional CNN [30] with 3M learnable parameters. The proposed *SSR-FCN* takes under 2 hours to train both Stage I and Stage II, and 4 miliseconds to predict a single (256×256) presentation attack/bonafide image on a Nvidia GTX 1080Ti GPU. In other words, *SSR-FCN* can process frames

(a) Paper Eyeglasses (b) Average Eyeglasses Score Map

Figure 2.10 A partial presentation attack artifact may be present in a small portion of the input 256×256 face image, such as (a) paper eyeglasses. However, since the proposed *SSR-FCN* dynamically aggregates decisions across multiple receptive fields in the image, a majority of the pixels in the final score map comprise of high scores (indicating the presence of a presentation attack). We visualize the average score map across all paper eyeglass attacks in (b).

at 250 Frames Per Second (FPS) and the size of the model is only 11.8MB. Therefore, *SSR-FCN* is well suited for deployment where real-time decisions are required.

2.6.10 Visualizing Presentation Attack Regions

SSR-FCN can automatically locate the individual presentation attack regions in an input face image. In Figure 2.9, we show heatmaps from the score maps extracted for a randomly chosen image from all presentation attack instruments. Red regions indicate a higher likelihood of presentation attack.

For a bonafide input image, the presentation attack regions are sparse with low likelihoods. In the case of replay and print attacks, the predicted presentation attack regions are located throughout the entire face image. This is because these presentation attacks contain global-level noise. For mask attacks, including half-mask, silicone mask, transparent mask, paper mask, and mannequin, the presentation attack patterns are identified near the eye and nose regions. Make-up attacks are

harder to detect since they are very subtle in nature. Proposed *SSR-FCN* detects obfuscation and cosmetic attack attempts by learning local discriminative cues around the eyebrow regions. In contrast, impersonation make-up patterns exist throughout the entire face. We also find that *SSR-FCN* can precisely locate the presentation attack artifacts, such as funny eyeglasses, paper glasses, and paper cut, in partial attacks.

2.7 Discussion

We show that the proposed *SSR-FCN* achieves superior generalization performance on SiW-M dataset [1] compared to the prevailing state-of-the-art methods that tend to overfit on the seen presentation attack instruments. Our method also achieves comparable performance to the state-of-the-art in Oulu-NPU dataset [2] and outperforms all baselines for cross-dataset generalization performance (CASIA-FASD [3] $\rightarrow$ Replay-Attack [4]).

In contrast to a number of prior studies [30, 88, 102, 123], the proposed approach does not utilize auxiliary cues for presentation attack detection, such as motion and depth information. While incorporating such cues may enhance performance on print and replay attack datasets such as Oulu-NPU, CASIA-MFSD, and Replay-Attack, it is at the risk of potentially overfitting to the two attacks and compute cost. A major benefit of *SSR-FCN* lies is its usability. A simple pre-processing step includes face detection and alignment. The cropped face is then passed to the network. With a *single* forward-pass through the FCN, we obtain both the score map and the final attack score.

Our proposed *SSR-FCN* outputs a bonafide/presentation attack decision at each pixel of intermediate feature maps. Due to the downsampling layers found after each convolutional operation (see Table 2.2), the decisions are automatically aggregated. Therefore, the final feature map (refered to as *score map*) is of much smaller resolution (16×16 pixels) compared to the original image. Therefore, in the case of partial attacks such as paper eyeglasses, even though a small portion of the face image comprises of a presentation attack artifact, our final score map is an aggregated

decision across multiple sliding windows (receptive field) of the original image. In Figure 2.10, we compute the average score map for all the paper eyeglasses presentation attacks. We find that, even though paper eyeglasses comprise of a small portion of the original image, majority of the pixels in the average *score map* comprise of high scores (indicating the presence of a presentation attack). In this paper, we obtain the final score via average pooling the score map. As future work, we intend on exploring other fusion mechanisms such as a weighted average via an attention mask.

Even though the proposed method is well-suited for generalizable face presentation attack detection, *SSR-FCN* is still limited by the amount and quality of available training data. For instance, when trained on a low-resolution dataset, namely Replay-Attack [4], cross-dataset generalization performance suffers.

2.8 Summary

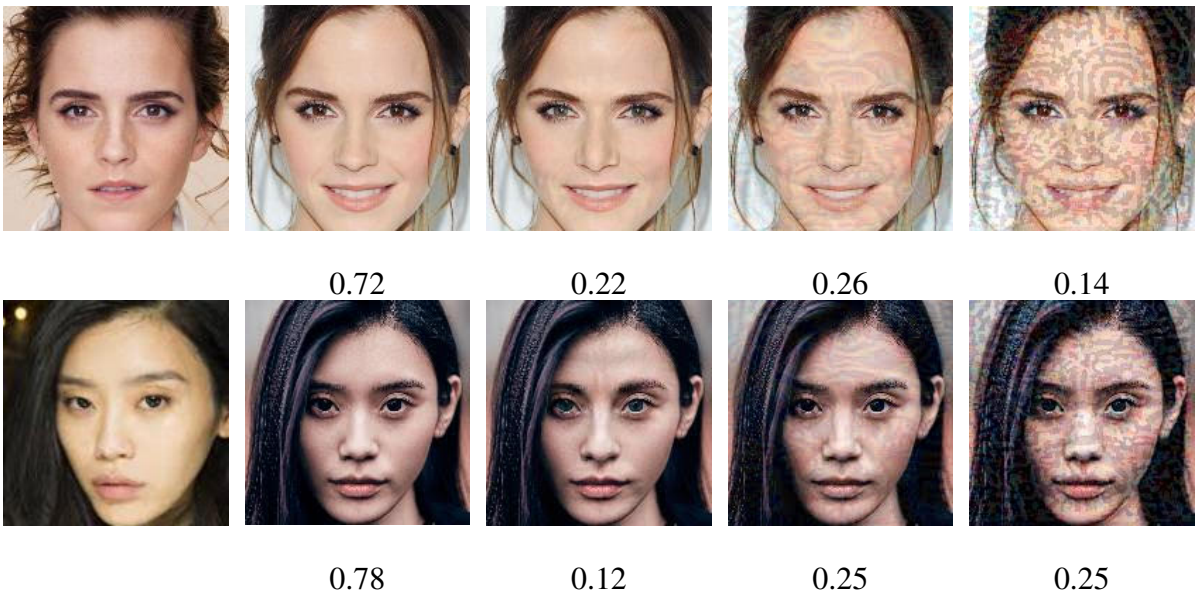
Face presentation attack detection systems are crucial for secure operation of an automated face recognition system. With the introduction of sophisticated presentation attacks, such as high resolution and tight fitting silicone 3D face masks, presentation attack detectors need to be robust and generalizable. We proposed a face presentation attack detection framework, namely *SSR-FCN*, that achieved state-of-the-art generalization performance against 13 different presentation attack instruments. *SSR-FCN* reduced the average error rate of competitive algorithms by 14% on one of the largest and most diverse face presentation attack detection dataset, SiW-M, comprised of 13 presentation attack instruments. It also generalizes well when training and testing datasets are from different sources. In addition, the proposed method is shown to be more interpretable compared to prior studies since it can directly predict the parts of the faces that are considered as presentation attacks. In the future, we intend on exploring whether incorporating domain knowledge in *SSR-FCN* can further improve generalization performance.

Chapter 3

Synthesizing and Defending Against Adversarial Faces

In the previous chapter, we proposed a solution to defend AFR systems against physically crafted face spoofs. However, in this chapter, we will show that prevailing AFR systems are also vulnerable to the growing threat of adversarial examples which are digital crafted. Our main focus is to first design an automatic adversarial synthesis method that can evade 5 state-of-the-art AFR systems. With a powerful adversarial face synthesizer, we evaluate the robustness of prevailing AFR systems against such digital adversarial attacks. The latter part of the chapter presents a state-of-the-art solution to safeguard AFR systems against any adversarial face.

3.1 Introduction



(a) Enrolled Face (b) Input Probe (c) AdvFaces (d) PGD [33] (e) FGSM [70]

Figure 3.1 Example gallery and probe face images and corresponding synthesized adversarial examples. (a) Two celebrities' real face photo enrolled in the gallery and (b) the same subject's probe image; (c) Adversarial examples generated from (b) by our proposed synthesis method, AdvFaces; (d-e) Results from two adversarial example generation methods. Cosine similarity scores ($\in [-1, 1]$) obtained by comparing (b-e) to the enrolled image in the gallery via ArcFace [9] are shown below the images. A score above **0.28** (threshold @ 0.1% False Accept Rate) indicates that two face images belong to the same subject. Here, a successful obfuscation attack would mean that humans can identify the adversarial probes and enrolled faces as belonging to the same identity but an automated face recognition system considers them to be from different subjects.

From mobile phone unlock, to boarding a flight at airports, the ubiquity of automated face recognition systems (AFR) is evident. With deep learning models, AFR systems are able to achieve accuracies as high as 99% True Accept Rate (TAR) at 0.1% False Accept Rate (FAR) [22]. The model behind this success is a Convolutional Neural Network (CNN) [6, 9, 44] and the availability of large face datasets to train the model. However, CNN models have been shown to be vulnerable to *adversarial perturbations*[1] [69–72]. Szegedy *et al.* first showed the dangers of *adversarial examples* in the image classification domain, where perturbing the pixels in the input image can cause CNNs to misclassify the image even when the amount of perturbation is imperceptible to

[1] Adversarial perturbations refer to altering an input image instance with small, human imperceptible changes in a manner that can evade CNN models.

(a) Print attack

(b) Replay attack

(c) Mask attack

(d) Adversarial Faces synthesized via proposed AdvFaces

Figure 3.2 Three types of face presentation attacks: (a) printed photograph, (b) replaying the targeted person's video on a smartphone, and (c) a silicone mask of the target's face. Face presentation attacks require a physical artifact. Adversarial attacks (d), on the other hand, are digital attacks that can compromise either a probe image or the gallery itself. To a human observer, face presentation attacks (a-c) are more conspicuous than adversarial faces (d).

the human eye [69]. Despite impressive recognition performance, prevailing AFR systems are still vulnerable to the growing threat of adversarial examples (see Figure 3.1).

To attack an AFR system, a hacker can maliciously perturb his face image in a manner that can cause AFR systems to match it to a target victim (*impersonation attack*) or any identity other than the hacker (*obfuscation attack*). Yet to the human observer, this adversarial face image should appear as a legitimate face photo of the attacker (see Figure 3.2d). This is different from face *presentation attacks*, where the hacker assumes the identity of a target by presenting a fake face (also known as spoof face) to a face recognition system (see Figure 3.2). However, in the case of presentation attacks, the hacker needs to actively participate by wearing a mask or replaying a photograph/video of the genuine individual which may be conspicuous in scenarios where human operators are involved (such as airports). As discussed below, adversarial faces, do not require active participation of the subject during authentication (comparison between adversarial probe and

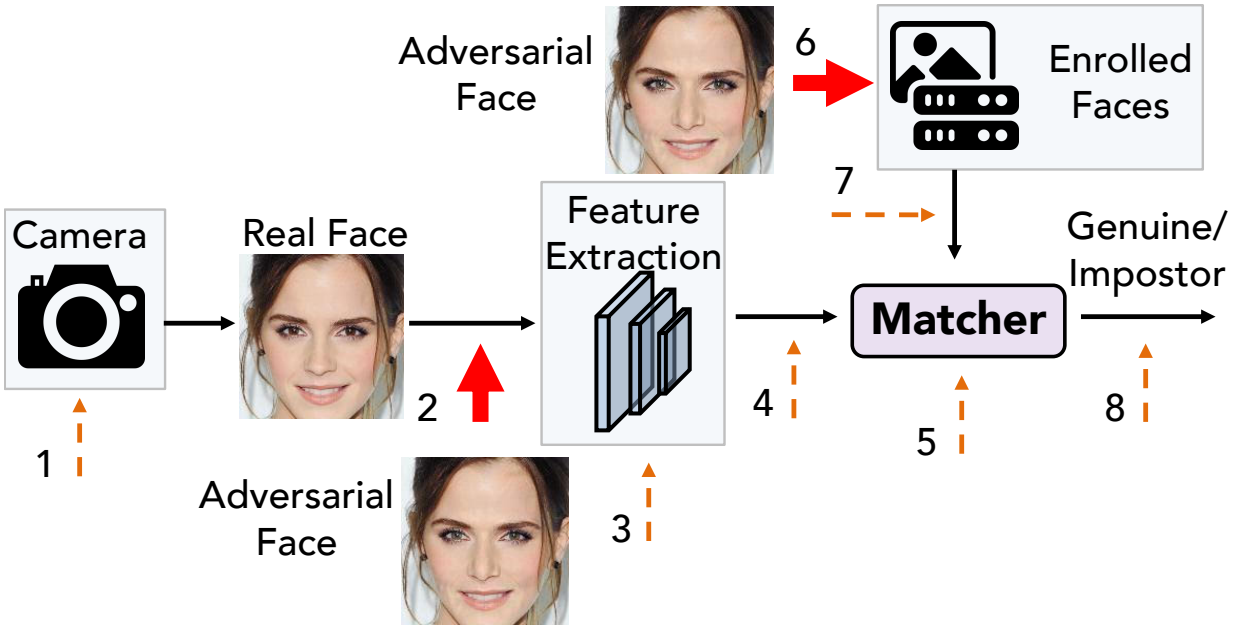


Figure 3.3 Eight points of attacks in an automated face recognition system [31]. An adversarial image can be injected in the AFR system at points 2 and 6 (solid arrows).

gallery images).

Consider for example, the United States Customs and Border Protection (CBP), the largest federal law enforcement agency in the United States [73], which (i) processes entry to the country for over a *million* travellers *everyday* [74] and (ii) employs automated face recognition for verifying travelers' identities [75]. In order to evade being identified as an individual in a CBP watchlist, a terrorist can maliciously enroll an adversarial image in the gallery such that upon entering the border, his legitimate face image will be matched to a known and benign individual or to a fake identity previously enrolled in the gallery. An individual can also generate adversarial examples to dodge his own identity in order to guard personal privacy. Ratha *et al.* [31] identified eight points in a biometric system where an attack can be launched against a biometric (including face) recognition system, including AFR (see Figure 3.3). An adversarial face image can be inserted in the AFR system at point 2, where compromised face embeddings will be obtained by the feature extractor that could be used for impersonation or obfuscation attacks. The entire gallery can also be compromised if the hacker enrolls an adversarial image at point 6, where none of the probes will match to the correct identity's gallery.

Three broad categories of adversarial attacks have been identified.

1. *White-box* attack: A majority of the prior work assumes full knowledge of the CNN model and then iteratively adds imperceptible perturbations to the probe image via various optimization schemes [32, 33, 70, 136–141]. This is unrealistic in real-world scenarios, since the attacker may not be able to access the models.
2. *Black-box* attack: Generally, black-box attacks are launched by querying the outputs of the deployed AFR system [142], [143]. But it may take a large number of queries to obtain a reasonable adversarial image [142]. Further, most Commercial-Off-The-Shelf (COTS) face matchers permit only a few queries at a time to prevent such attacks.
3. *Semi-whitebox* attack: Here, a white-box model is utilized *only during training* and then adversarial examples are synthesized during inference without any knowledge of the deployed AFR model.

This chapter first proposes a GAN-based adversarial face synthesis method, namely *AdvFaces*, that learns to generate visually realistic adversarial face images that are misclassified by state-of-the-art AFR systems. The latter part of the chapter utilizes the concepts learned from AdvFaces in order to defend AFR sytems against any adversarial attack type.

3.2 Related Work

3.2.1 Generative Adversarial Networks (GANs)

Generative Adversarial Networks [144] have been shown to be successful in a wide variety of image synthesis applications [145, 146] such as style transfer [147–149], image-to-image translation [150, 151], and representation learning [145, 152, 153]. Our objective is to synthesize face images that are not only visually realistic but are also able to evade AFR systems.

3.2.2 Adversarial Attacks on Image Classification

Majority of the published papers have focused on white-box attacks, where the hacker has full access to the model that is being attacked [33, 69, 70, 136, 137]. Other works focused on optimizing adversarial perturbation by minimizing an objective function for targeted attacks while satisfying certain constraints [136]. However, these white-box approaches are not feasible in the face recognition domain, as the attacker is unlikely to have access to the deployed AFR system. We propose a feed-forward network that can automatically generate an adversarial image with a single forward pass without the need for any knowledge of AFR system during inference.

Indeed, feed-forward networks have been used for synthesizing adversarial attacks. Baluja and Fischer proposed a deep autoencoder that learns to transform an input image to an adversarial image [154]. Studies on synthesizing adversarial instances via GANs are limited in literature [155–157]. These methods require softmax probabilities in order to evade an image classifier. Instead, we propose an identity loss function better suited for generating adversarial faces using the face embeddings obtained from a face matcher.

3.2.3 Adversarial Attacks on Face Recognition

In literature, studies on generating adversarial examples in the face recognition domain are relatively limited. Bose *et al.* craft adversarial examples by solving constrained optimization such that a face detector cannot detect a face [158]. In [159, 160], perturbations are constrained to the eyeglass region of the face and adversarial image is generated by gradient-based methods. However, these methods rely on white-box manipulations of face recognition models, which is impractical in real-world scenarios. Dong *et al.* proposed an evolutionary optimization method for generating adversarial faces in black-box settings [142]. However, they require at least 1,000 queries to the target AFR system before a realistic adversarial face can be synthesized. Song *et al.* employed a conditional variation autoencoder GAN for crafting adversarial face images in a semi-whitebox setting [161]. However, they only focused on impersonation attacks and require at least 5 images of the target subject for training and inference. In contrast, we train a GAN that can perform both

	Study	Method	Dataset	Attacks	Self-Sup.
Robustness	Adv. Training [11] (2017)	Train with adv.	ImageNet [162]	FGSM [34]	×
	RobGAN [12] (2019)	Train with generated adv.	CIFAR10 [163], ImageNet [162]	PGD [35]	×
	Feat. Denoising [164] (2019)	Custom network arch.	ImageNet [162]	PGD [35]	×
	L2L [13] (2019)	Train with generated adv.	MNIST [165], CIFAR10 [163]	FGSM [34], PGD [35], C&W [136]	✓
Detection	Gong *et al.* [166] (2017)	Binary CNN	MNIST [165], CIFAR10 [163]	FGSM [34]	×
	UAP-D [167] (2018)	PCA+SVM	MEDS [168], MultiPIE [169], PaSC [170]	UAP [171]	×
	SmartBox [172] (2018)	Adaptive Noise	Yale Face [173]	DeepFool [141], EAD [174], FGSM [34]	×
	ODIN [175] (2018)	Out-of-distribution Detection	CIFAR10 [163], ImageNet [162]	OOD samples	×
	Goswami *et al.* [176] (2019)	SVM on AFR Filters	MEDS [168], PaSC [170], MBGC [177]	Black-box, EAD [174]	×
	Steganalysis [178] (2019)	Steganlysis	ImageNet [162]	FGSM [34], DeepFool [141], C&W [136]	×
	Massoli *et al.* [179] (2020)	MLP/LSTM on AFR Filters	VGGFace2 [180]	BIM [181], FGSM [34], C&W [136]	×
	Agarwal *et al.* [182] (2020)	Image Transformation	ImageNet [162], MBGC [177]	FGSM [34], PGD [35], DeepFool [141]	×
Purification	MagNet [14] (2017)	AE Purifier	MNIST [165], CIFAR10 [163]	FGSM [34], DeepFool [141], C&W [136]	×
	DefenseGAN [15] (2018)	GAN	MNIST [165], CIFAR10 [163]	FGSM [34], C&W [136]	×
	Feat. Distillation [183] (2019)	JPEG-compression	MNIST [165], CIFAR10 [163]	FGSM [34], DeepFool [141], C&W [136]	×
	NRP [184] (2020)	AE Purifier	ImageNet [162]	FGSM [34]	✓
	A-VAE [185] (2020)	Variational AE	LFW [8]	FGSM [34], PGD [35], C&W [136]	×
	FaceGuard (this study)	Adv. Generator + Detector + Purifier	LFW [8], Celeb-A [17], FFHQ [18]	FGSM [34], PGD [35], DeepFool [141], AdvFaces [16], GFLM [32], Semantic [186]	✓

Table 3.1 Related work in adversarial defenses used as baselines in our study. Unlike majority of prior work, *FaceGuard* is self-supervised where no pre-computed adversarial examples are required for training.

obfuscation and impersonation attacks and requires a single face image of the target subject.

3.2.4 Defenses Against Adversarial Attacks

In literature, a common defense strategy, namely *robustness* is to re-train the classifier we wish to defend with adversarial examples [11, 13, 34, 35]. However, adversarial training has been shown to degrade classification accuracy on real (non-adversarial) images [187, 188].

In order to prevent degradation in AFR performance, a large number of adversarial defense mechanisms are deployed as a pre-processing step, namely *detection*, which involves training a binary classifier to distinguish between real and adversarial examples [166, 167, 172, 179, 189–

199]. The attacks considered in these studies [200–203] were initially proposed in the object recognition domain and they often fail to detect the attacks in a feature-extraction network setting, as in face recognition. Therefore, prevailing detectors against adversarial faces are demonstrated to be effective only in a highly constrained setting where the number of subjects is limited and fixed during training and testing [167, 172, 179].

Another pre-processing strategy, namely *purification*, involves automatically removing adversarial perturbations in the input image prior to passing them to a face matcher [14, 15, 184, 204]. However, without a dedicated adversarial detector, these defenses may end up "purifying" a real face image, resulting in high false reject rates.

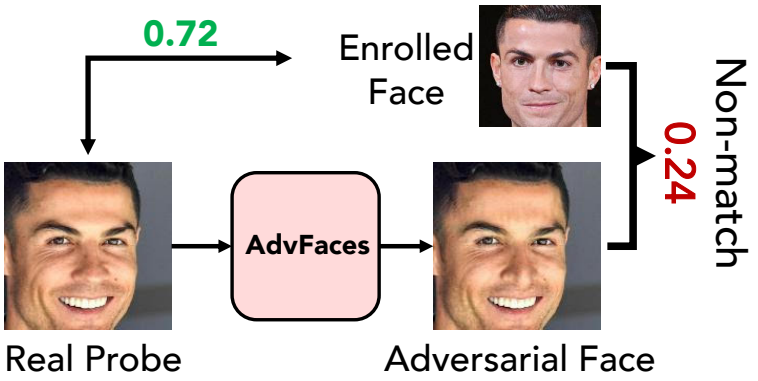
In Tab. 3.1, we summarize a few studies on adversarial defenses that are used as baselines in our work.

3.3 Synthesizing Adversarial Faces

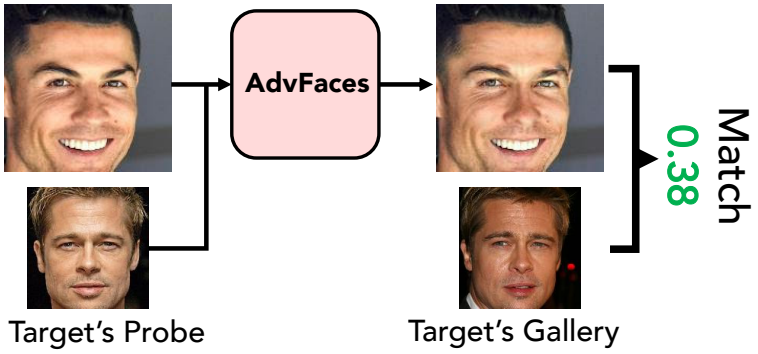
Semi-whitebox settings are more appropriate for crafting adversarial faces; once the network learns to generate the perturbed instances based on a single face recognition system, attacks can be transferred to any black-box AFR systems. However, past approaches, based on Generative Adversarial Networks (GANs) [155–157], were proposed in the image classification domain and rely on softmax probabilities [155–157, 161, 205]. Therefore, the number of object classes are assumed to be known during training and testing. Face recognition systems do not utilize the softmax layer for classification (as the number of identities are not fixed) instead features from the last fully connected layer are used for comparing face images. Approaches for crafting adversarial faces include adding makeup, eyeglasses, hat, or occlusions to faces [159–161, 205, 206].

We emphasize the following requirements of the adversarial face generator:

- Adversarial face images should be perceptually realistic such that a human observer can identify the image as a legitimate face image.
- The faces need to be perturbed in a manner such that they cannot be identified as the hacker



(a) Obfuscation Attack



(b) Impersonation Attack

Figure 3.4 Once trained, AdvFaces automatically generates an adversarial face image. During an obfuscation attack, (a) the adversarial face appears to be a benign example of Cristiano Ronaldo's face, however, it fails to match his enrolled image. AdvFaces can also combine Cristiano's probe and Brad Pitt's probe to synthesize an adversarial image that looks like Cristiano but matches Brad's gallery image (b).

(*obfuscation attack*) or automatically matched to a target subject (*impersonation attack*) by an AFR system.

- The amount of perturbation should be controllable by the hacker so that he can examine the success of the learning model as a function of amount of perturbation.
- The adversarial examples should be *transferable* and *model-agnostic* (*i.e.*treat the target AFR model as a black-box). In other words, the generated adversarial examples should have high attack success rate on other black-box AFR systems as well.

We propose an automated adversarial face synthesis method, named *AdvFaces*, which generates an adversarial image for a probe face image and satisfies all the above requirements (see Fig. 3.4).

The contributions of the paper are as follows:

1. GAN-based AdvFaces that learns to generate visually realistic adversarial face images that are misclassified by state-of-the-art AFR systems.
2. Adversarial faces generated via AdvFaces are model-agnostic and transferable, and achieve high success rate on 5 state-of-the-art automated face recognition systems.
3. Perceptual studies where human observers suggest that the adversarial examples appear similar to the probe.
4. Visualizing the facial regions, where pixels are perturbed and analyzing the transferability of AdvFaces.
5. An open-source[2] automated adversarial face generator permitting users to control the amount of perturbation.

3.3.1 Proposed Methodology

Our goal is to synthesize a face image that visually appears to pertain to the target face, yet automatic face recognition systems either incorrectly matches the synthesized image to another person or does not match to target's gallery images. AdvFaces comprises of a generator $\mathcal{G}$, a discriminator $\mathcal{D}$, and face matcher (see Figure 3.5).

Generator The proposed generator takes an input face image, $\mathbf{x} \in \mathcal{X}$, and outputs an image, $\mathcal{G}(\mathbf{x})$. The generator is conditioned on the input image $\mathbf{x}$; for different input faces, we will get different synthesized images.

Since our goal is to obtain an adversarial image that is metrically similar to the probe in the image space, $\mathbf{x}$, it is not desirable to perturb all the pixels in the probe image. For this reason, we treat the output from the generator as an additive mask and the adversarial face is defined as $\mathbf{x} + \mathcal{G}(\mathbf{x})$. If the magnitude of the pixels in $\mathcal{G}(\mathbf{x})$ is minimal, then the adversarial image comprises mostly of the probe $\mathbf{x}$. Here, we denote $\mathcal{G}(\mathbf{x})$ as an "adversarial mask". In order to bound the

[2]https://github.com/ronny3050/AdvFaces

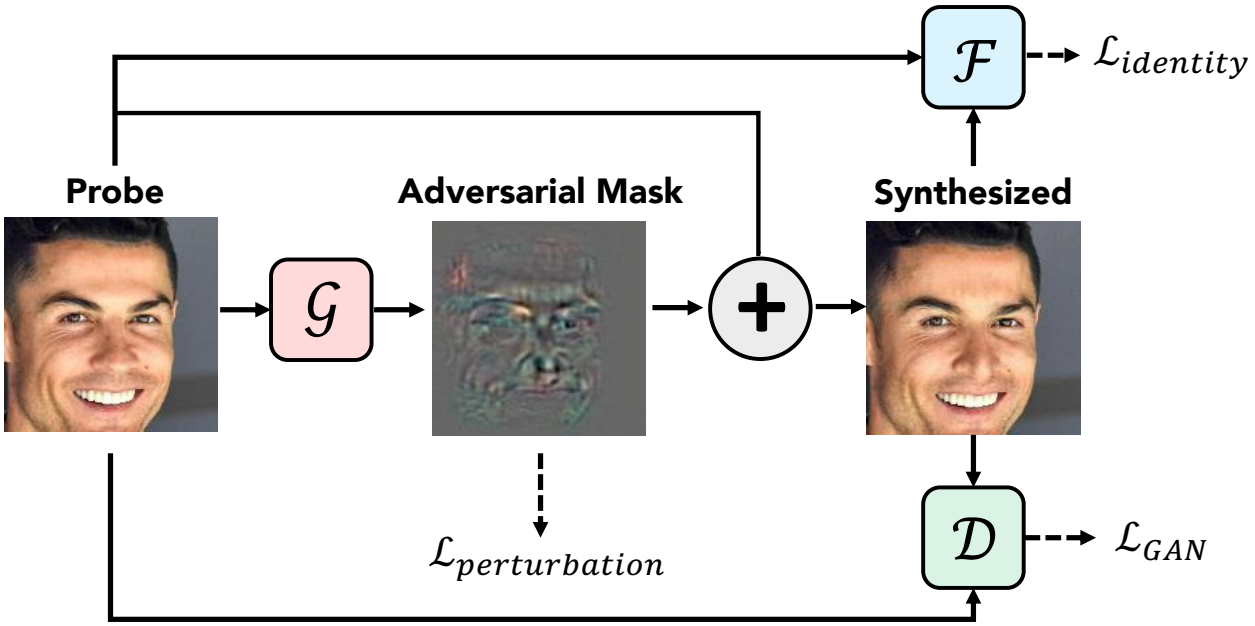


Figure 3.5 Given a probe face image, AdvFaces automatically generates an adversarial mask that is then added to the probe to obtain an adversarial face image.

magnitude of the adversarial mask, we introduce a *perturbation loss* during training by minimizing the L_2 norm[3]:

$$\mathcal{L}_{perturbation} = \mathbb{E}_{\mathbf{x}} \left[\max \left(\epsilon, \|\mathcal{G}(\mathbf{x})\|_2 \right) \right] \tag{3.3.1}$$

where $\epsilon \in [0, \infty)$ is a hyperparameter that controls the minimum amount of perturbation allowed.

In order to achieve our goal of impersonating a target subject's face or obfuscating one's own identity, we need a face matcher, $\mathcal{F}$, to supervise the training of AdvFaces. For obfuscation attack, at each training iteration, AdvFaces tries to minimize the cosine similarity between face embeddings of the input probe $\mathbf{x}$ and the generated image $\mathbf{x} + \mathcal{G}(\mathbf{x})$ via an *identity* loss function:

$$\mathcal{L}_{identity} = \mathbb{E}_{\mathbf{x}}[\mathcal{F}(\mathbf{x}, \mathbf{x} + \mathcal{G}(\mathbf{x}))] \tag{3.3.2}$$

For an impersonation attack, AdvFaces maximizes the cosine similarity between the face embeddings of a randomly chosen target's probe, $\mathbf{y}$, and the generated adversarial face $\mathbf{x} + \mathcal{G}(\mathbf{x})$ via:

$$\mathcal{L}_{identity} = \mathbb{E}_{\mathbf{x}}[1 - \mathcal{F}(\mathbf{y}, \mathbf{x} + \mathcal{G}(\mathbf{x}))] \tag{3.3.3}$$

[3]For brevity, we denote $\mathbb{E}_{\mathbf{x}} \equiv \mathbb{E}_{\mathbf{x} \in \mathcal{X}}$.

Obfuscation Attack	AdvFaces	GFLM [32]	PGD [33]	FGSM [70]
Attack Success Rate (%) @ 0.1% FAR				
□ FaceNet [6]	99.67	23.34	99.70	**99.96**
■ SphereFace [44]	97.22	29.49	**99.34**	98.71
■ ArcFace [9]	**64.53**	03.43	33.25	35.30
■ COTS-A	**82.98**	08.89	18.74	32.48
■ COTS-B	**60.71**	05.05	01.49	18.75
Structural Similarity	**0.95 ± 0.01**	0.82 ± 0.12	0.29 ± 0.06	0.25 ± 0.06
Computation Time (s)	**0.01**	3.22	11.74	0.03

Impersonation Attack	AdvFaces	A^3GN [161]	PGD [33]	FGSM [70]
Attack Success Rate (%) @ 0.1% FAR				
□ FaceNet [6]	20.85 ± 0.40	05.99 ± 0.19	**76.79 ± 0.26**	13.04 ± 0.12
■ SphereFace [44]	**20.19 ± 0.27**	07.94 ± 0.19	09.03 ± 0.39	02.34 ± 0.03
■ ArcFace [9]	**24.30 ± 0.44**	17.14 ± 0.29	19.50 ± 1.95	08.34 ± 0.21
■ COTS-A	**20.75 ± 0.35**	15.01 ± 0.30	01.76 ± 0.10	01.40 ± 0.08
■ COTS-B	**19.85 ± 0.28**	10.23 ± 0.50	12.49 ± 0.24	04.67 ± 0.16
Structural Similarity	**0.92 ± 0.02**	0.69 ± 0.04	0.77 ± 0.04	0.48 ± 0.75
Computation Time (s)	**0.01**	0.04	11.74	0.03

□ White-box matcher (used for training) ■ Black-box matcher (never used in training)

Table 3.2 Attack success rates and structural similarities between probe and gallery images for obfuscation and impersonation attacks. Attack rates for obfuscation comprises of 484,514 comparisons and the mean and standard deviation across 10-folds for impersonation reported. The mean and standard deviation of the structural similarities between adversarial and probe images along with the time taken to generate a single adversarial image (on a Quadro M6000 GPU) also reported.

The perturbation and identity loss functions enforce the network to learn the salient facial regions that can be perturbed minimally in order to evade automatic face recognition systems.

Discriminator Akin to previous works on GANs [144, 150], we introduce a discriminator in order to encourage perceptual realism of the generated images. We use a fully-convolution network as a patch-based discriminator [150]. Here, the discriminator, $\mathcal{D}$, aims to distinguish between a probe, $\mathbf{x}$, and a generated adversarial face image $\mathbf{x} + \mathcal{G}(\mathbf{x})$ via a GAN loss:

$$\begin{aligned} \mathcal{L}_{GAN} = \quad & \mathbb{E}_{\mathbf{x}}\left[\log \mathcal{D}(\mathbf{x})\right] + \\ & \mathbb{E}_{\mathbf{x}}[\log(1 - \mathcal{D}(\mathbf{x} + \mathcal{G}(\mathbf{x})))] \end{aligned} \tag{3.3.4}$$

Finally, AdvFaces is trained in an end-to-end fashion with the following objectives:

$$\min_{\mathcal{D}} \mathcal{L}_{\mathcal{D}} = -\mathcal{L}_{GAN} \tag{3.3.5}$$

$$\min_{\mathcal{G}} \mathcal{L}_{\mathcal{G}} = \mathcal{L}_{GAN} + \lambda_i \mathcal{L}_{identity} + \lambda_p \mathcal{L}_{perturbation} \tag{3.3.6}$$

where λ_i and λ_p are hyper-parameters controlling the relative importance of identity and perturbation losses, respectively. Note that $\mathcal{L}_{GAN}$ and $\mathcal{L}_{perturbation}$ encourage the generated images to be visually similar to the original face images, while $\mathcal{L}_{identity}$ optimizes for a high attack success rate. After training, the generator $\mathcal{G}$ can generate an adversarial face image for any input image and can be tested on any black-box face recognition system.

3.3.2 Experimental Settings

Evaluation Metrics We quantify the effectiveness of the adversarial attacks generated by AdvFaces and other state-of-the-art baselines via (i) *attack success rate* and (ii) *structural similarity (SSIM)*.

The attack success rate for *obfuscation attack* is computed as,

$$\text{Attack Success Rate} = \frac{(\text{No. of Comparisons} < \tau)}{\text{Total No. of Comparisons}} \tag{3.3.7}$$

where each comparison consists of a subject's adversarial probe and an enrollment image. Here, τ is a pre-determined threshold computed at, say, 0.1% FAR[4]. Attack success rate for *impersonation attack* is defined as,

$$\text{Attack Success Rate} = \frac{(\text{No. of Comparisons} \geq \tau)}{\text{Total No. of Comparisons}} \tag{3.3.8}$$

Here, a comparison comprises of an adversarial image synthesized with a target's probe and matched to the target's enrolled image. We evaluate the success rate for the impersonation setting via 10-fold cross-validation where each fold consists of a randomly chosen target.

Similar to prior studies [161], in order to measure the similarity between the adversarial ex-

[4]For each face matcher, we pre-compute the threshold at 0.1% FAR on all possible image pairs in LFW. For *e.g.*, threshold @ 0.1% FAR for ArcFace is 0.28.

ample and the input face, we compute the structural similarity index (SSIM) between the images. SSIM is a normalized metric between -1 (completely different image pairs) to 1 (identical image pairs).

Datasets We train AdvFaces on CASIA-WebFace [10] and then test on LFW [8][5].

- **CASIA-WebFace** [10] is comprised of 494,414 face images belonging to 10,575 different subjects. We removed 84 subjects that are also present in LFW and the testing images in this chapter.
- **LFW** [8] contains 13,233 web-collected images of 5,749 different subjects. In order to compute the attack success rate, we only consider subjects with at least two face images. After this filtering, 9,614 face images of 1,680 subjects are available for evaluation.

All the testing images in this chapter have no identity overlap with the training set, CASIA-WebFace [10].

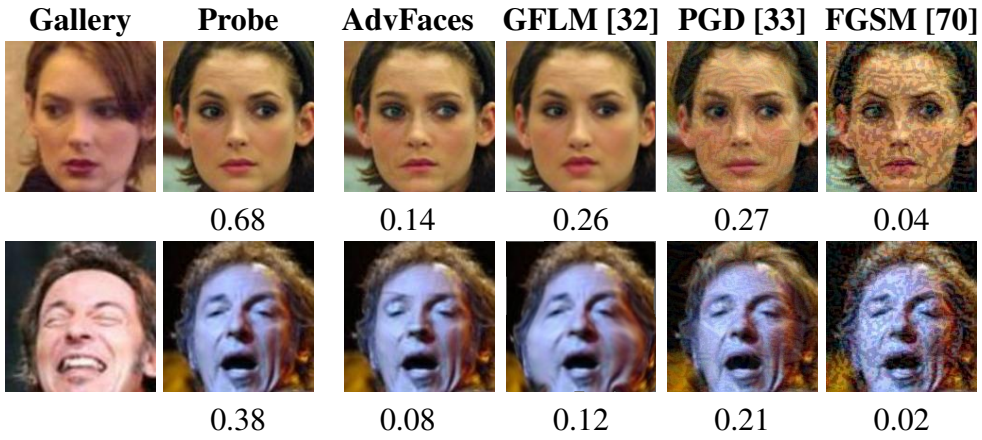
Experimental Settings We use ADAM optimizers in Tensorflow with $\beta_1 = 0.5$ and $\beta_2 = 0.9$ for the entire network. Each mini-batch consists of 32 face images. We train AdvFaces for 200,000 steps with a fixed learning rate of 0.0001. Since our goal is to generate adversarial faces with high success rate, the identity loss is of utmost importance. We empirically set $\lambda_i = 10.0$ and $\lambda_p = 1.0$. We train two separate models and set $\epsilon = 3.0$ and $\epsilon = 8.0$ for obfuscation and impersonation attacks, respectively. Architecture details are provided in the Addendum (Sec. 3.3.9).

Face Recognition Systems For all our experiments, we employ 5 state-of-the-art face matchers[6]. Three of them are publicly available, namely, FaceNet [6], SphereFace [44], and ArcFace [9]. We also report our results on two commercial-off-the-shelf (COTS) face matchers, COTS-A and COTS-B[7]. We use FaceNet [6] as the white-box face recognition model, $\mathcal{F}$, during training. *All*

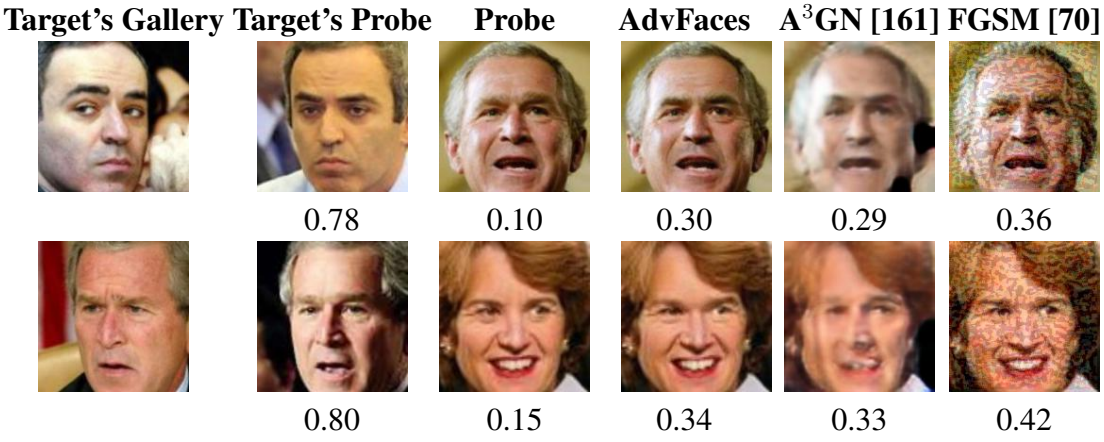
[5]Training on CASIA-WebFace and evaluating on LFW is a common approach in face recognition literature [9, 44]

[6]All the open-source and COTS matchers achieve 99% accuracy on LFW under LFW protocol.

[7]Both COTS-A and COTS-B utilize CNNs for face recognition. COTS-B is one of the top performers in the NIST Ongoing Face Recognition Vendor Test (FRVT) [207].



(a) Obfuscation Attack



(b) Impersonation Attack

Figure 3.6 Adversarial face synthesis results on LFW dataset in (a) obfuscation and (b) impersonation attack settings (cosine similarity scores obtained from ArcFace [9] with threshold @ 0.1% FAR$= 0.28$). The proposed method synthesizes adversarial faces that are seemingly inconspicuous and maintain high perceptual quality. Additional examples are available in the Addendum (Sec. 3.3.9).

the testing images in this chapter are generated from the same model (trained only with FaceNet) and tested on different matchers.

3.3.3 Comparison with State-of-the-Art

We compare our adversarial face synthesis method with state-of-the-art methods that have specifically been implemented or proposed for faces, including GFLM [32], PGD [33], FGSM [70],

and A^{3}GN [161][8]. In Table 3.2, we find that compared to the state-of-the-art, AdvFaces generates adversarial faces that are similar to the probe 3.6. Moreover, the adversarial images attain a high obfuscation attack success rate on 4 state-of-the-art black-box AFR systems in both obfuscation and impersonation settings. AdvFaces learns to perturb the salient regions of the face, unlike PGD [33] and FGSM [70], which alter every pixel in the image. GFLM [32], on the other hand, geometrically warps the face images and thereby, results in low structural similarity. In addition, the state-of-the-art matchers are robust to such geometric deformation which explains the low success rate of GFLM on face matchers. A^{3}GN, another GAN-based method, however, fails to achieve a reasonable success rate in an impersonation setting.

3.3.4 Ablation Study

In order to analyze the importance of each module in our system, in Figure 3.7, we train three variants of AdvFaces for comparison by removing the discriminator ($\mathcal{D}$), perturbation loss $\mathcal{L}_{perturbation}$, and identity loss $\mathcal{L}_{identity}$, respectively. The discriminator helps to ensure the visual quality of the synthesized faces are maintained. With the generator alone, undesirable artifacts are introduced. Without the proposed perturbation loss, perturbations in the adversarial mask are unbounded and therefore, leads to a lack in perceptual quality. The identity loss is imperative in ensuring an adversarial image is obtained. Without the identity loss, the synthesized image cannot evade state-of-the-art face matchers. We find that every component of AdvFaces is necessary in order to obtain an adversarial face that is not only perceptually realistic but can also evade state-of-the-art face matchers.

3.3.5 What is AdvFaces Learning?

Via $\mathcal{L}_{perturbation}$, during training, AdvFaces learns to perturb only the salient facial regions that can evade the face matcher, $\mathcal{F}$ (FaceNet [6] in our case). In Figure 3.8, AdvFaces synthesizes the adversarial masks corresponding to the probes. We then threshold the mask to extract pixels with

[8]We train the baselines using their official implementations (detailed in the Addendum (Sec. 3.3.9)).

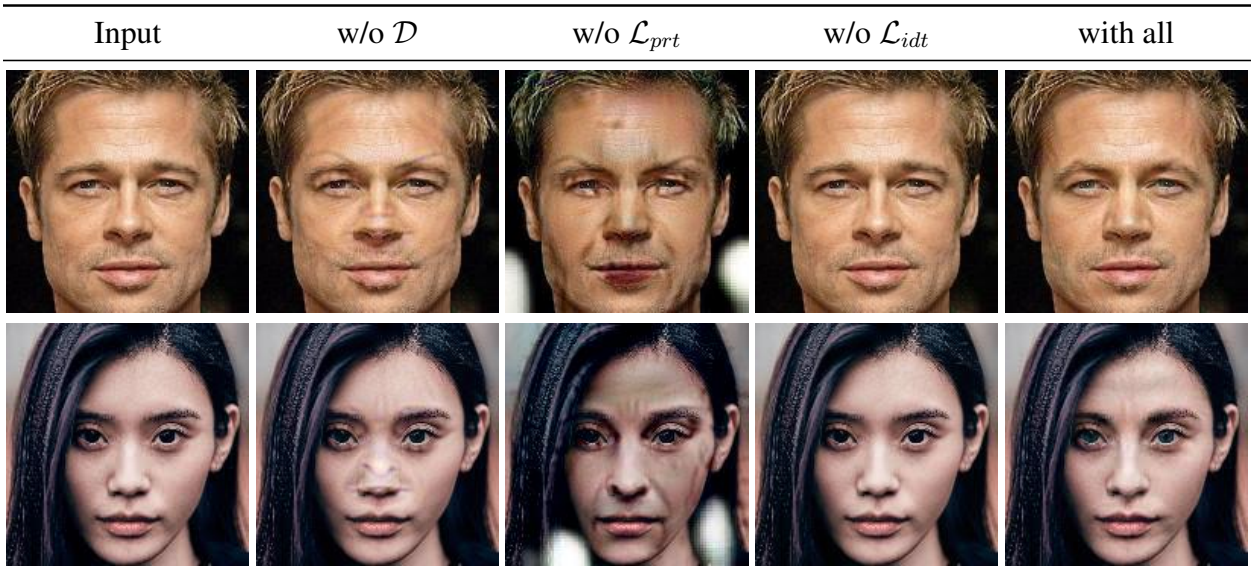


Figure 3.7 Variants of AdvFaces trained without the discriminator, perturbation loss, and identity loss, respectively. Every component of AdvFaces is necessary.

perturbation magnitudes exceeding 0.40. It can be inferred that the eyebrows, eyeballs, and nose contain highly discriminative information that an AFR system utilizes to identify an individual. Therefore, perturbing these salient regions are enough to evade state-of-the-art face recognition systems.

3.3.6 Transferability of AdvFaces

In Table 3.2, we find that attacks synthesized by AdvFaces when trained on a white-box matcher (FaceNet), can successfully evade 5 other face matchers that are not utilized during training in both obfuscation and impersonation settings. In order to investigate the transferability property of AdvFaces, we extract face embeddings of real images and their corresponding adversarial images, under the obfuscation setting, via the white-box matcher (FaceNet) and a black-box matcher (ArcFace). In total, we extract feature vectors from 1,456 face images of 10 subjects in the LFW dataset [8]. In Figure 3.9, we plot the correlation heatmap between face features of real images, their corresponding adversarial masks and adversarial images. First, we observe that face embeddings of real images extracted by FaceNet and ArcFace are correlated in a similar fashion.

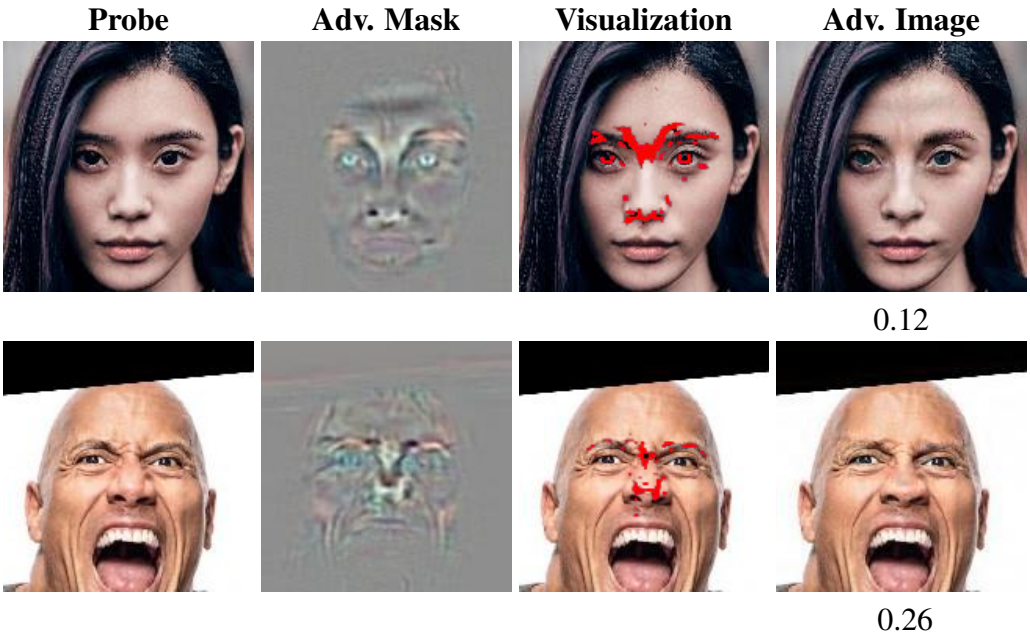


Figure 3.8 State-of-the-art face matchers can be evaded by slightly perturbing salient facial regions, such as eyebrows, eyeballs, and nose (cosine similarity obtained via ArcFace [9]).

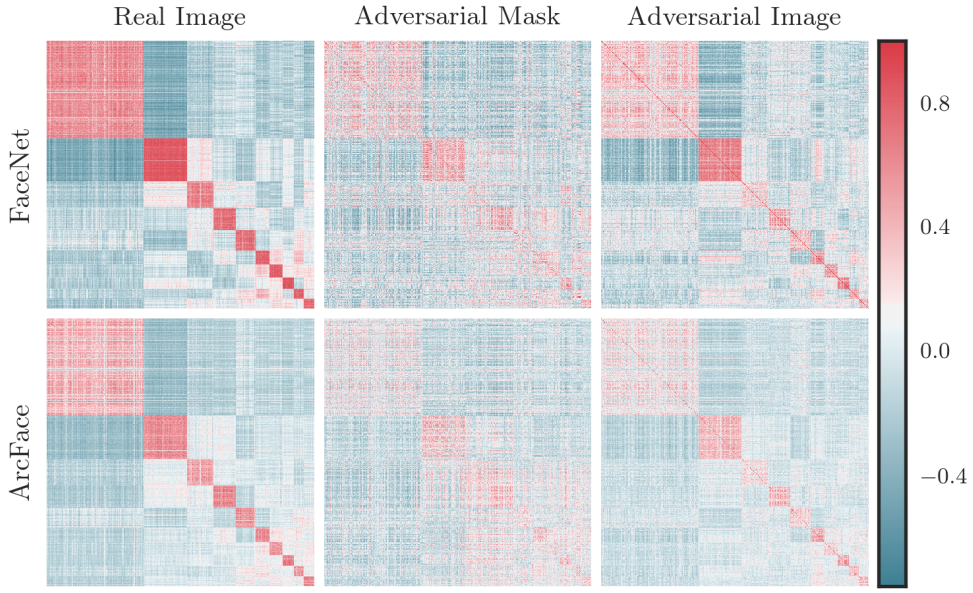


Figure 3.9 Correlation between face features extracted via FaceNet and ArcFace from 1,456 images belonging to 10 subjects.

This indicates that both matchers extract features with related pairwise correlations. Consequently, perturbing salient features for FaceNet can lead to high attack success rates for ArcFace as well. The similarity among the correlation distributions of both matchers can also be observed when

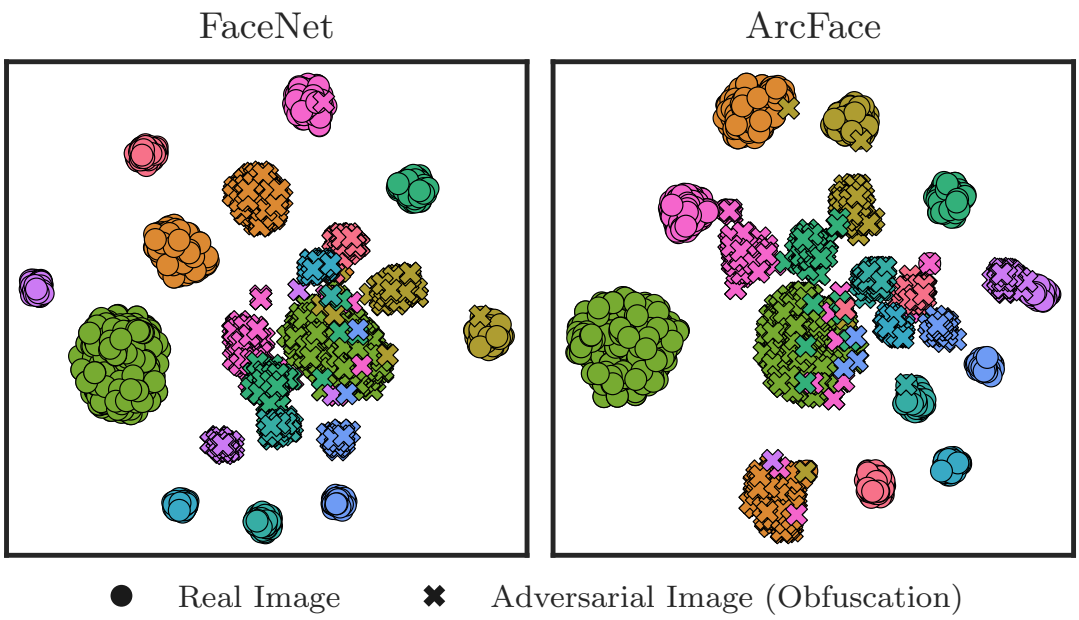


Figure 3.10 2D t-SNE visualization of face representations extracted via FaceNet and ArcFace from 1,456 images belonging to 10 subjects.

adversarial masks and adversarial images are input to the matchers. That is, receptive fields for automatic face recognition systems attend to similar regions in the face. To further illustrate the distributions of the embeddings of real and synthesized images, we plot the 2D t-SNE visualization of the face embeddings for the 10 subjects in Figure 3.10. The identity clusterings can be clearly observed from both real and adversarial images. In particular, the adversarial counterpart of each subject forms a new cluster that draws closer to the adversarial clusterings of other subjects. This shows that AdvFaces perturbs only salient pixels related to face identity while maintaining a semantic meaning in the feature space, resulting in a similar manifold of synthesized faces to that of real faces.

3.3.7 Effect of Perturbation Amount

The perturbation loss, $\mathcal{L}_{perturbation}$ is bounded by a hyper-parameter, ϵ, *i.e.*, the L_2 norm of the adversarial mask must be at least ϵ. Without this constraint, the adversarial mask becomes a blank image with no changes to the probe. With ϵ, we can observe a trade-off between the attack success

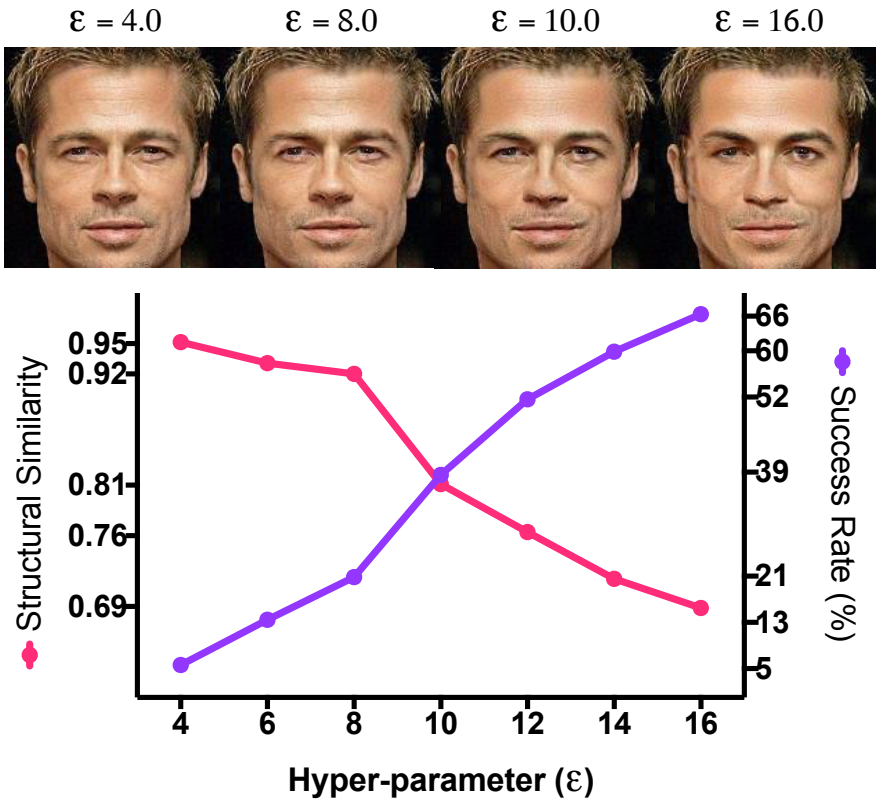
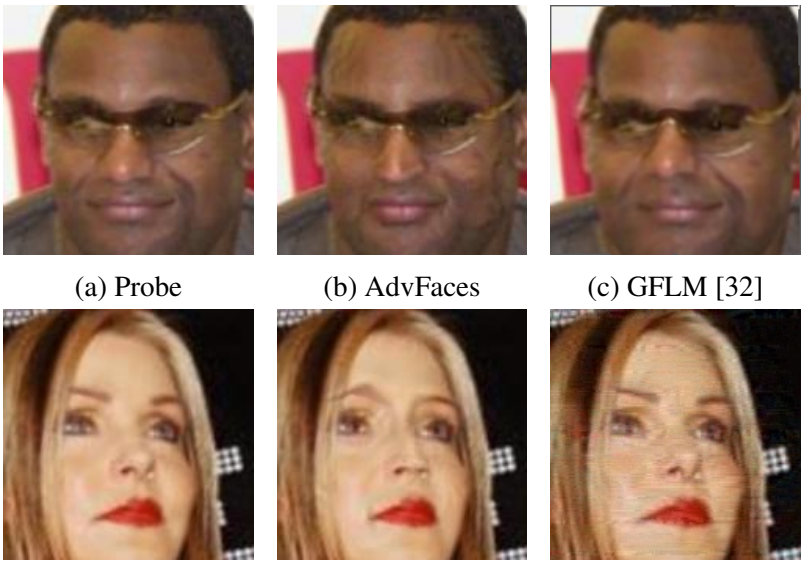


Figure 3.11 Trade-off between attack success rate and structural similarity for impersonation attacks. We choose $\epsilon = 8.0$.

rate and the structural similarity between the probe and synthesized adversarial face (Fig. 3.11). A higher ϵ leads to less perturbation restriction, resulting in a higher attack success rate at the cost of a lower structural similarity. For an impersonation attack, this implies that the adversarial image may contain facial features from both the hacker and the target. In our experiments, we chose $\epsilon = 8.0$ and $\epsilon = 3.0$ for impersonation and obfuscation attacks, respectively.

3.3.8 Human Perceptual Study

For 500 real face images (probes), we generate 500 corresponding adversarial examples via AdvFaces, GFLM [32], A^{3}GN [161], PGD [33], and FGSM [70]. We then performed a user study on Amazon Mechanical Turk (AMT). A worker is shown a probe along with the 5 adversarial faces. The worker then has unlimited time to decide which adversarial face, among the 5 possible choices, is the most similar to the probe.



(a) Probe (b) AdvFaces (c) GFLM [32]

(d) Probe (e) AdvFaces (f) PGD [33]

Figure 3.12 Example failure cases where human observers voted an adversarial image synthesized by (c) GFLM [32] and (f) PGD [33] to be closer to the probe face (a), (d) compared to AdvFaces (b), (e).

Table 3.3 For each method, the average and standard deviation (%) of the number of times workers chose the synthesized image to be closest to the probe.

Method	**Hit Rate (%)**
AdvFaces	**62.06 ± 5.06**
GFLM [32]	23.52 ± 3.82
A^3GN [161]	00.60 ± 0.84
PGD [33]	12.80 ± 3.88
FGSM [70]	00.38 ± 0.73

In total, we compute results from 100 different workers. From Tables 3.2 and 3.3, we find that AdvFaces generates adversarial faces that are not only effective in evading face matchers, but are also visually similar to the probes and outperforms the state-of-the-art adversarial face synthesis methods. Indeed, some adversarial face images synthesized by the baselines are voted to be closer to the probe (see Figure 3.12), however, compared to AdvFaces, these methods have a low success rate (see Table 3.2).

3.3.9 Addendum

3.3.9.1 Implementation Details

AdvFaces is implemented using Tensorflow r1.12.0. A single NVIDIA Quadro M6000 GPU is used for training and testing.

Data Preprocessing All face images are passed through MTCNN face detector [41] to detect five landmarks (two eyes, nose, and two mouth corners). Via similarity transformation, the face images are aligned. After transformation, the images are resized to 160×160. Prior to training and testing, each pixel in the RGB image is normalized by subtracting 127.5 and dividing by 128.

Architecture Let `c7s1-k` be a 7×7 convolutional layer with k filters and stride 1. `dk` denotes a 4×4 convolutional layer with k filters and stride 2. `Rk` denotes a residual block that contains two 3×3 convolutional layers. `uk` denotes a $2\times$ upsampling layer followed by a 5×5 convolutional layer with k filters and stride 1. We apply Instance Normalization and Batch Normalization to the generator and discriminator, respectively. We use Leaky ReLU with slope 0.2 in the discriminator and ReLU activation in the generator. The architectures of the two modules are as follows:

- Generator:

  `c7s1-64,d128,d256,R256,R256,R256, u128, u64, c7s1-3`

- Discriminator:

  `d32,d64,d128,d256,d512`

A 1×1 convolutional layer with 3 filters and stride 1 is attached to the last convolutional layer of the discriminator for the patch-based GAN loss $\mathcal{L}_{GAN}$.

We apply the `tanh` activation function on the last convolution layer of the generator to ensure that the generated image $\in [-1, 1]$. In the chapter, we denoted the output of the tanh layer as an "adversarial mask", $\mathcal{G}(x) \in [-1, 1]$ and $x \in [-1, 1]$. The final adversarial image is computed as $x_{adv} = 2 \times \texttt{clamp}\left[\mathcal{G}(x) + \left(\frac{x+1}{2}\right)\right]_0^1 - 1$. This ensures $\mathcal{G}(x)$ can either add or subtract pixels from x when $\mathcal{G}(x) \neq 0$. When $\mathcal{G}(x) \to 0$, then $x_{adv} \to x$.

The overall algorithm describing the training procedure of *AdvFaces* can be found in Algorithm 1.

Algorithm 1 Training *AdvFaces*. All experiments in this work use $\alpha = 0.0001$, $\beta_1 = 0.5$, $\beta_2 = 0.9$, $\lambda_i = 10.0$, $\lambda_p = 1.0$, $m = 32$.
We set $\epsilon = 3.0$ (obfuscation), $\epsilon = 8.0$ (impersonation).

Input
  X Training Dataset
  $\mathcal{F}$ Cosine similarity between an image pair obtained by face matcher
  $\mathcal{G}$ Generator with weights $\mathcal{G}_\theta$
  $\mathcal{D}$ Discriminator with weights $\mathcal{D}_\theta$
  m Batch size
  α Learning rate
for number of training iterations **do**
  Sample a batch of probes $\{x^{(i)}\}_{i=1}^{m} \sim \mathcal{X}$
  if impersonation attack **then**
    Sample a batch of target images $y^{(i)} \sim \mathcal{X}$
    $\delta^{(i)} = \mathcal{G}((x^{(i)}, y^{(i)})$
  else if obfuscation attack **then**
    $\delta^{(i)} = \mathcal{G}(x^{(i)})$
  end if
  $x_{adv}^{(i)} = x^{(i)} + \delta^{(i)}$
  $\mathcal{L}_{perturbation} = \frac{1}{m}\left[\sum_{i=1}^{m} \max\left(\epsilon, ||\delta^{(i)}||_2\right)\right]$
  if impersonation attack **then**
    $\mathcal{L}_{identity} = \frac{1}{m}\left[\sum_{i=1}^{m} \mathcal{F}\left(x^{(i)}, x_{adv}^{(i)}\right)\right]$
  else if obfuscation attack **then**
    $\mathcal{L}_{identity} = \frac{1}{m}\left[\sum_{i=1}^{m} \left(1 - \mathcal{F}\left(y^{(i)}, x_{adv}^{(i)}\right)\right)\right]$
  end if
  $\mathcal{L}_{GAN}^{\mathcal{G}} = \frac{1}{m}\left[\sum_{i=1}^{m} log\left(1 - \mathcal{D}(x_{adv}^{(i)})\right)\right]$
  $\mathcal{L}^{\mathcal{D}} = \frac{1}{m}\sum_{i=1}^{m}\left[log\left(\mathcal{D}(x^{(i)})\right) + log\left(1 - \mathcal{D}(x_{adv}^{(i)})\right)\right]$
  $\mathcal{L}^{\mathcal{G}} = \mathcal{L}_{GAN}^{\mathcal{G}} + \lambda_i \mathcal{L}_{identity} + \lambda_p \mathcal{L}_{perturbation}$
  $\mathcal{G}_\theta = \texttt{Adam}(\nabla_{\mathcal{G}}\mathcal{L}^{\mathcal{G}}, \mathcal{G}_\theta, \alpha, \beta_1, \beta_2)$
  $\mathcal{D}_\theta = \texttt{Adam}(\nabla_{\mathcal{D}}\mathcal{L}^{\mathcal{D}}, \mathcal{D}_\theta, \alpha, \beta_1, \beta_2)$
end for

3.3.9.2 Effect on Cosine Similarity

In Figure 3.13 we see the effect on cosine similarity scores when adversarial face images synthesized by AdvFaces is introduced to a black-box face matcher, ArcFace [9]. A majority (64.53%)

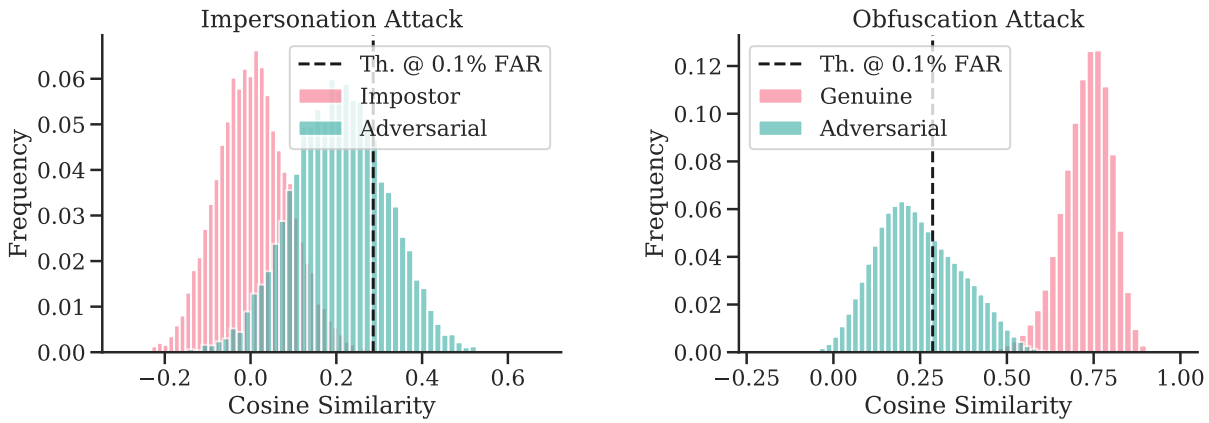


Figure 3.13 Shift in cosine similarity scores for ArcFace [9] before and after adversarial attacks generated via AdvFaces.

of the scores fall below the threshold at 0.1% FAR causing the AFR system to falsely reject under obfuscation attack. In the impersonation attack setting, the system falsely accepts 24.30% of the image pairs.

3.3.9.3 Structural Similarity

Image comparison techniques, such Mean Squared Error (MSE) or Peak Signal-to-Noise Ratio (PSNR), estimate the absolute errors, disregarding the *perceptual* differences; on the other hand, SSIM is a perception-based model that considers image differences as perceived change in structural information, while also incorporating important perceptual phenomena, including both luminance masking and contrast masking terms. For instance, consider the image pair comprising of two images of Ming Xi. We can notice that perceptually, the image pairs are similar, but this perceptual similarity is not reflected appropriately in MSE and PSNR. Since, SSIM is a normalized similarity metric, it is better suited for our application where a face image pair is subjectively judged by human operators.

(a) Probe (b) Adversarial

SSIM: 00.96 MSE: 40.82 PSNR: 32.02

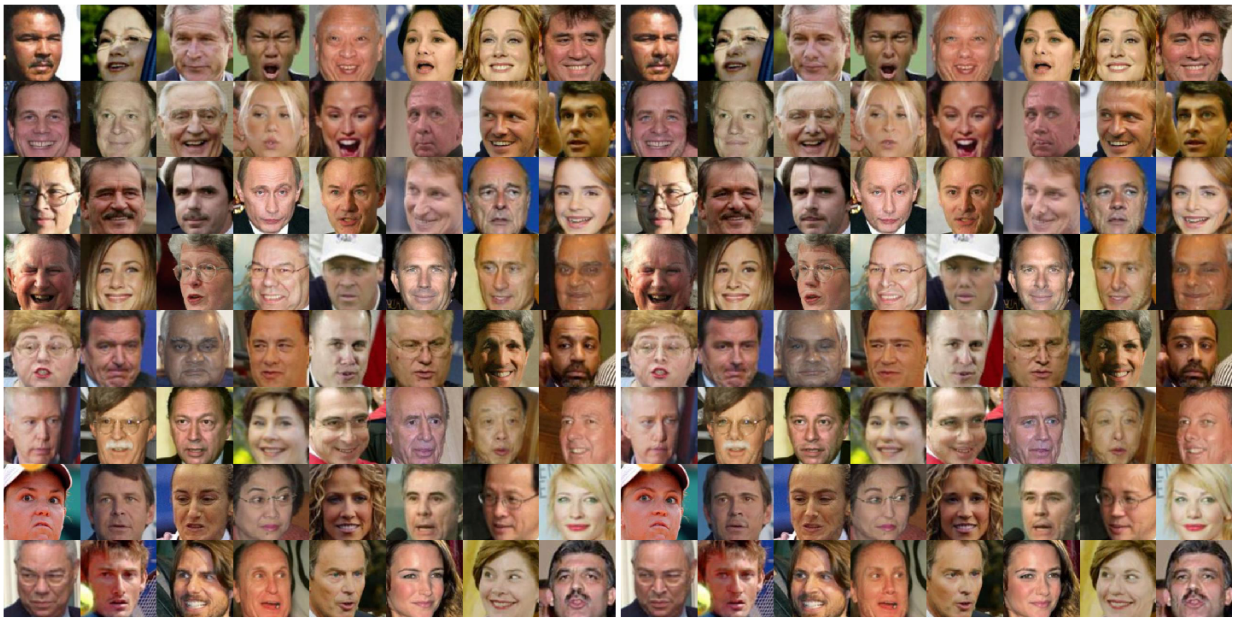


Figure 3.15 Left: Real face images in the LFW dataset. Right: Adversarial images synthesized via AdvFaces under obfuscation setting.

3.3.9.4 Baseline Implementation Details

All the state-of-the-art baselines in the chapter are implementations proposed specifically for evading face recognition systems.

FGSM [70] We use the Cleverhans implementation[9] of FGSM on FaceNet. This implementation supports both obfuscation and impersonation attacks. The only modification was changing $\epsilon = 0.01$ to $\epsilon = 0.08$ in order to create more effective attacks.

PGD [33] We use a variant of PGD proposed specifically for face recognition systems[10]. Originally, this implementation is proposed for impersonation attacks, however, for obfuscation we randomly choose a target other than genuine subject. We do not make any modifications to the parameters.

GFLM [32] Code for this landmark-based attack synthesis method is publicly available[11]. This method relies on softmax probalities implying that the training and testing identities are fixed. Originally, the classifier is trained on CASIA-WebFace. However, for a fairer evaluation, we trained a face classifier on LFW and then ran the attack.

A^{3}GN [161] To the best of our knowledge, there is no publicly available implementation of A^{3}GN. We made the following modifications to achieve an effective baseline:

- The authors originally used ArcFace [9] as the target model. Since all other baselines employ FaceNet as the target model, we also used FaceNet for training A^{3}GN.
- Originally, a cycle-consistency loss was proposed for content preservation. However, we were not able to reproduce this and therefore, opted for the same L_1 norm loss, but without the second generator. This greatly helps in the visual quality of the generated adversarial image. That is, we modified Equation 3 [161], from $\mathcal{L}_{rec} = \mathbb{E}_{x,z}\left[||x - G_2(G_1(x,z))||_1\right]$ to $\mathcal{L}_{rec} = \mathbb{E}_{x,z}\left[||x - G_1(x,z)||_1\right]$

[9] https://github.com/tensorflow/cleverhans/tree/master/examples/facenet_adversarial_faces
[10] https://github.com/ppwwyyxx/Adversarial-Face-Attack
[11] https://github.com/alldbi/FLM

3.4 Defending Against Adversarial Faces

The accuracy, usability, and touchless acquisition of state-of-the-art (SOTA) AFR systems have led to their ubiquitous adoption in a plethora of domains. However, this has also inadvertently sparked a community of attackers that dedicate their time and effort to manipulate faces either physically [208, 209] or digitally [210], in order to evade AFR systems [82]. AFR systems have been shown to be vulnerable to adversarial attacks resulting from perturbing an input probe [16, 32, 142, 186]. Even when the amount of perturbation is imperceptible to the human eye, such adversarial attacks can degrade the face recognition performance of SOTA AFR systems [16]. With the growing dissemination of "fake news" and "deepfakes" [81], research groups and social media platforms alike are pushing towards generalizable defense against continuously evolving adversarial attacks.

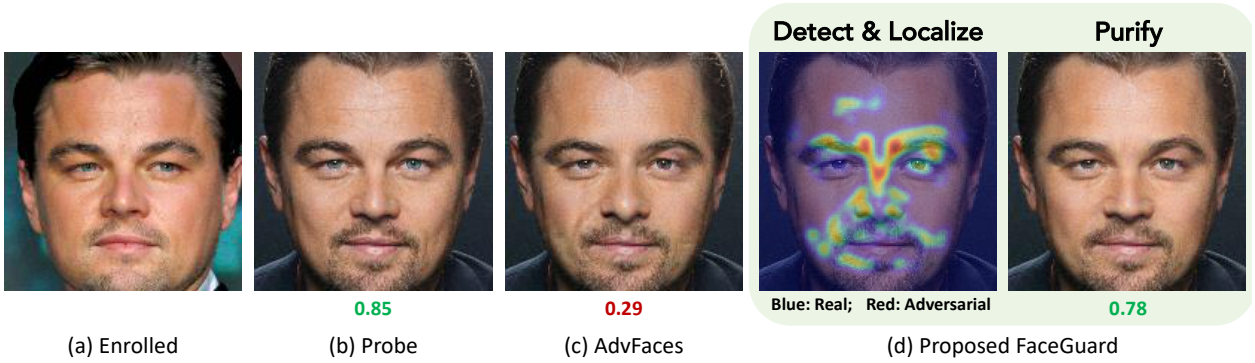


Figure 3.16 Leonardo DiCaprio's real face photo (a) enrolled in the gallery and (b) his probe image[12]; (c) Adversarial probe synthesized by a state-of-the-art (SOTA) adversarial face generator, AdvFaces [16]; (d) Proposed adversarial defense framework, namely *FaceGuard* takes (c) as input, detects adversarial images, localizes perturbed regions, and outputs a "purified" face devoid of adversarial perturbations. A SOTA face recognition system, ArcFace, fails to match Leonardo's adversarial face (c) to (a), however, the purified face can successfully match to (a). Cosine similarity scores ($\in [-1, 1]$) obtained via ArcFace [9] are shown below the images. A score above **0.36** (threshold @ 0.1% False Accept Rate) indicates that two faces are of the same subject.

A considerable amount of research has focused on synthesizing adversarial attacks [16, 32, 34, 35, 141, 186]. Obfuscation attempts (faces are perturbed such that they cannot be identified as the attacker) are more effective [16], computationally efficient to synthesize [34, 35], and widely

[11] https://bit.ly/2IkfSxk

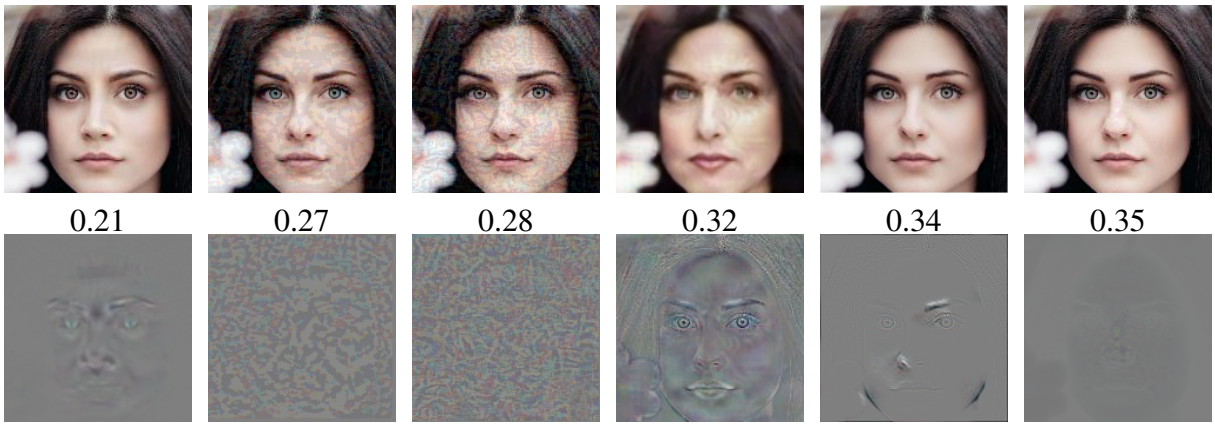


Figure 3.17 *(Top Row)* Adversarial faces synthesized via 6 adversarial attacks used in our study. *(Bottom Row)* Corresponding adversarial perturbations (gray indicates no change from the input). Notice the diversity in the perturbations. ArcFace scores between adversarial image and the unaltered gallery image (not shown here) are given below each image. A score above **0.36** indicates that two faces are of the same subject. Zoom in for details.

adopted [211] compared to impersonation attacks (perturbed faces can automatically match to a target subject). Similar to prior defense efforts [172, 179], this section of the chapter focuses on defending against obfuscation attacks (see Fig. 3.16). Given an input probe image, $\mathbf{x}$, an adversarial generator has two requirements under the obfuscation scenario: (1) synthesize an adversarial face image, $\mathbf{x}_{adv} = \mathbf{x} + \delta$, such that SOTA AFR systems fail to match $\mathbf{x}_{adv}$ and $\mathbf{x}$, and (2) limit the magnitude of perturbation $||\delta||_p$ such that $\mathbf{x}_{adv}$ appears very similar to $\mathbf{x}$ to humans.

A number of approaches have been proposed to defend against adversarial attacks. Their major shortcoming is *generalizability* to unseen adversarial attacks. Adversarial face perturbations may vary significantly (see Fig. 3.17). For instance, gradient-based attacks, such as FGSM [35] and PGD [35], perturb every pixel in the face image, whereas, AdvFaces [16] and SemanticAdv [186] perturb only the salient facial regions, *e.g.*, eyes, nose, and mouth. On the other hand, GFLM [32] performs geometric warping to the face. Since the exact type of adversarial perturbation may not be known a priori, a defense system trained on a subset of adversarial attack types may have degraded performance on other unseen attacks.

To the best of our knowledge, we take the first step towards a complete defense against adversarial faces by integrating an adversarial face generator, a detector, and a purifier into a unified

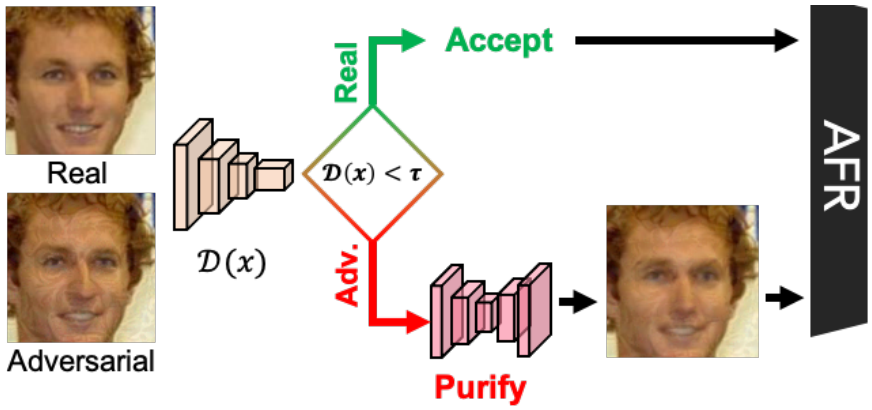
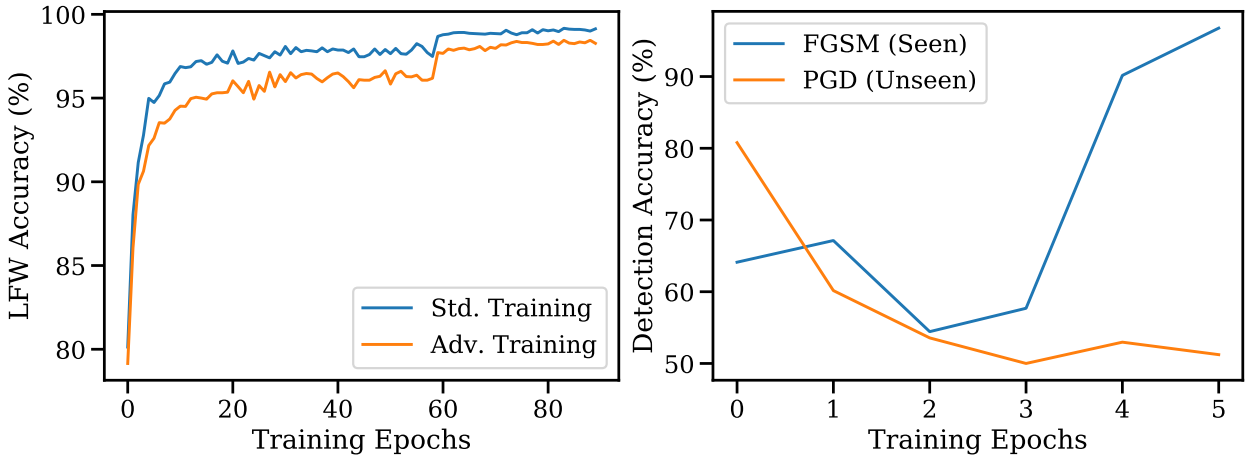


Figure 3.18 *FaceGuard* employs a detector ($\mathcal{D}$) to compute an adversarial score. Scores below detection threshold (τ) passes the input to AFR, and high value invokes a purifier and sends the purified face to the AFR system.

framework, namely *FaceGuard* (see Fig. 3.18). Robustness to unseen adversarial attacks is imparted via a stochastic generator that outputs diverse perturbations evading an AFR system, while a detector continuously learns to distinguish them from real faces. Concurrently, a purifier removes the adversarial perturbations from the synthesized image.

This work makes the following contributions:

- A new self-supervised framework, namely *FaceGuard*, for defending against adversarial face images. *FaceGuard* combines benefits of adversarial training, detection, and purification into a unified defense mechanism trained in an end-to-end manner.
- With the proposed diversity loss, a generator is regularized to produce stochastic and challenging adversarial faces. We show that the diversity in output perturbations is sufficient for improving *FaceGuard*'s robustness to unseen attacks compared to utilizing pre-computed training samples from known attacks.
- Synthesized adversarial faces aid the detector to learn a tight decision boundary around real faces. *FaceGuard*'s detector achieves SOTA detection accuracies of 99.81%, 98.73%, and 99.35% on 6 unseen attacks on LFW [8], Celeb-A [17], and FFHQ [18].
- As the generator trains, a purifier concurrently removes perturbations from the synthesized adversarial faces. With the proposed bonafide loss, the detector also guides purifier's training to ensure purified images are devoid of adversarial perturbations. At 0.1% False Accept



(a) Adversarial Training [11] (b) Detection [166]

Figure 3.19 (a) Adversarial training degrades AFR performance of FaceNet matcher [6] on real faces in LFW dataset compared to standard training. (b) A binary classifier trained to distinguish between real faces and FGSM [34] attacks fails to detect unseen attack type, namely PGD [35].

Rate, *FaceGuard*'s purifier enhances the True Accept Rate of ArcFace [9] from 34.27% under no defense to 77.46%.

3.4.1 Limitations of State-of-the-Art Defenses

Robustness. Adversarial training is regarded as one of the most effective defense method [12, 34, 35] on small datasets including MNIST and CIFAR10. Whether this technique can scale to large datasets and a variety of different attack types (perturbation sets) has not yet been shown. Adversarial training is formulated as [34, 35]:

$$\min_{\theta} \underset{(x,y)\sim\mathcal{P}_{data}}{\mathbb{E}} \left[\max_{\delta\in\Delta} \ell\left(f_\theta\left(x+\delta\right), y\right) \right], \tag{3.4.1}$$

where $(x, y) \sim \mathcal{P}_{data}$ is the (image, label) joint distribution of data, $f_\theta(x)$ is the network parameterized by θ, and $\ell\left(f_\theta\left(x\right), y\right)$ is the loss function (usually cross-entropy). Since the ground truth data distribution, $\mathcal{P}_{data}$, is not known in practice, it is later replaced by the empirical distribution. Here, the network, f_θ is made robust by training with an adversarial noise (δ) that maximally increases the classification loss. In other words, adversarial training involves training with the *strongest* adversarial attack.

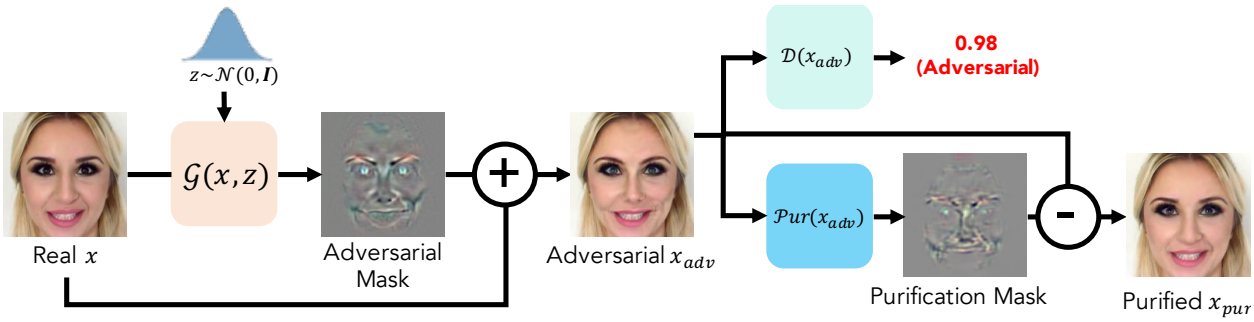


Figure 3.20 Overview of training the proposed *FaceGuard* in a self-supervised manner. An *adversarial generator*, $\mathcal{G}$, continuously learns to synthesize challenging and diverse perturbations that evade a face matcher. At the same time, a *detector*, $\mathcal{D}$, learns to distinguish between the synthesized adversarial faces and real face images. Perturbations residing in the synthesized adversarial faces are removed via a *purifier*, $\mathcal{P}ur$.

The generalization of adversarial training has been in question [12, 13, 187, 188, 212]. It was shown that adversarial training can significantly reduce classification accuracy on real examples [187, 188]. In the context of face recognition, we illustrate this by training two face matchers on CASIA-WebFace: (i) FaceNet [6] trained via the standard training process, and (ii) FaceNet [6] by adversarial training (FGSM[13]). We then compute face recognition performance across training iterations on a separate testing dataset, LFW [8]. Fig. 3.19a shows that adversarial training drops the accuracy from $99.13\% \rightarrow 98.27\%$. We gain the following insight: adversarial training may degrade AFR performance on real faces.

Detection. Detection-based approaches employ a pre-processing step to "detect" whether an input face is real or adversarial [166, 167, 179, 191]. A common approach is to utilize a binary classifier, $\mathcal{D}$, that maps a face image, $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ to $\{0, 1\}$, where 0 indicates a real and 1 an adversarial face. We train a binary classifier to distinguish between real and FGSM attack samples in CASIA-WebFace [10]. In Fig. 3.19b, we evaluate its detection accuracy on FGSM and PGD samples in LFW [8]. We find that prevailing detection-based defense schemes may overfit to the specific adversarial attacks utilized for training.

[13]With max perturbation hyperparameter as $\epsilon = 8/256$.

3.4.2 Proposed Methodology

Our defense aims to achieve robustness without sacrificing AFR performance on real face images. We posit that an adversarial defense trained alongside an adversarial generator in a *self-supervised* manner may improve robustness to unseen attacks. The main intuitions behind our defense mechanism are as follows:

- Since adversarial training may degrade AFR performance, we opt to obtain a robust adversarial *detector* and *purifier* to detect and purify adversarial attacks.
- Given that prevailing detection-based methods tend to overfit to known adversarial perturbations (see Addendum (Sec. 3.4.6)), a detector and purifier trained on *diverse* synthesized adversarial perturbations may be more robust to unseen attacks.
- Sufficient diversity in synthesized perturbations can guide the detector to learn a tighter boundary around real faces. In this case, the detector itself can serve as a powerful supervision for the purifier.
- Lastly, pixels involved in the purification process may serve to indicate adversarial regions in the input face.

3.4.2.1 Adversarial Generator

The generalizability of an adversarial detector and purifier relies on the quality of the synthesized adversarial face images output by *FaceGuard*'s adversarial generator. We propose an adversarial generator that continuously learns to synthesize challenging and diverse adversarial face images.

The generator, denoted as $\mathcal{G}$, takes an input real face image, $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$, and outputs an adversarial perturbation $\mathcal{G}(\mathbf{x}, \mathbf{z})$, where $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ is a random latent vector. Inspired by prevailing adversarial attack generators [16, 34, 35, 136, 141], we treat the output perturbation $\mathcal{G}(\mathbf{x}, \mathbf{z})$ as an additive *perturbation mask*. The final adversarial face image, $\mathbf{x}_{adv}$, is given by $\mathbf{x}_{adv} = \mathbf{x} + \mathcal{G}(\mathbf{x}, \mathbf{z})$.

In an effort to impart generalizability to the detector and purifier, we emphasize the following requirements of $\mathcal{G}$:

- **Adversarial:** Perturbatation, $\mathcal{G}(\mathbf{x}, \mathbf{z})$, needs to be adversarial such that an AFR system can-

not identify the adversarial face image $\mathbf{x}_{adv}$ as the same person as the input probe $\mathbf{x}$.

- **Visually Realistic:** Perturbation $\mathcal{G}(\mathbf{x}, \mathbf{z})$ should also be minimal such that $\mathbf{x}_{adv}$ appears as a legitimate face image of the subject in the input probe $\mathbf{x}$.
- **Stochastic:** For an input $\mathbf{x}$, we require diverse adversarial perturbations, $\mathcal{G}(\mathbf{x}, \mathbf{z})$, for different latents $\mathbf{z}$.

For satisfying all of the above requirements, we propose multiple loss functions to train the generator.

Obfuscation Loss To ensure $\mathcal{G}(\mathbf{x}, \mathbf{z})$ is indeed *adversarial*, we incorporate a white-box AFR system, $\mathcal{F}$, to supervise the generator. Given an input face, $\mathbf{x}$, the generator aims to output an adversarial face, $\mathbf{x}_{adv} = \mathbf{x} + \mathcal{G}(\mathbf{x}, \mathbf{z})$ such that the face representations, $\mathcal{F}(\mathbf{x})$ and $\mathcal{F}(\mathbf{x}_{adv})$, do not match. In other words, the goal is to minimize the cosine similarity between the two face representations[14]:

$$\mathcal{L}_{obf} = \mathbb{E}_{\mathbf{x}} \left[\frac{\mathcal{F}(\mathbf{x}) \cdot \mathcal{F}(\mathbf{x}_{adv})}{||\mathcal{F}(\mathbf{x})||\, ||\mathcal{F}(\mathbf{x}_{adv})||} \right]. \tag{3.4.2}$$

Perturbation Loss With the identity loss alone, the generator may output perturbations with large magnitudes which will (a) be trivial for the detector to reject and (b) violate the visual realism requirement of x_{adv}. Therefore, we restrict the perturbations to be within $[-\epsilon, \epsilon]$ via a hinge loss:

$$\mathcal{L}_{pt} = \mathbb{E}_{\mathbf{x}} \left[\max \left(\epsilon, ||\mathcal{G}(\mathbf{x}, \mathbf{z})||_2 \right) \right]. \tag{3.4.3}$$

Diversity Loss The above two losses jointly ensure that at each step, our generator learns to output challenging adversarial attacks. However, these attacks are deterministic; for an input image, we will obtain the same adversarial image. This may again lead to an inferior detector that overfits to a few deterministic perturbations seen during training. Motivated by studies of preventing mode collapse in GANs [213], we propose maximizing a diversity loss to promote stochastic perturbations per training iteration, i:

$$\mathcal{L}_{div} = -\frac{1}{N_{ite}} \sum_{i=1}^{N_{ite}} \frac{\left|\left| \mathcal{G}(\mathbf{x}, \mathbf{z}_1)^{(i)} - \mathcal{G}(\mathbf{x}, \mathbf{z}_2)^{(i)} \right|\right|_1}{||\mathbf{z}_1 - \mathbf{z}_2||_1}, \tag{3.4.4}$$

where N_{ite} is the number of training iterations, $\mathcal{G}(\mathbf{x}, \mathbf{z})^{(i)}$ is the perturbation output at iteration i,

[14]For brevity, we denote $\mathbb{E}_{\mathbf{x}} \equiv \mathbb{E}_{\mathbf{x} \in \mathcal{P}_{data}}$.

and $(\mathbf{z}_1, \mathbf{z}_2)$ are two i.i.d. samples from $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$. The diversity loss ensures that for two random latent vectors, $\mathbf{z}_1$ and $\mathbf{z}_2$, we will obtain two different perturbations $\mathcal{G}(\mathbf{x}, \mathbf{z}_1)^{(i)}$ and $\mathcal{G}(\mathbf{x}, \mathbf{z}_2)^{(i)}$.

GAN Loss Akin to prior work on GANs [144, 150], we introduce a discriminator to encourage perceptual realism of the adversarial images. The discriminator, Dsc, aims to distinguish between probes, $\mathbf{x}$, and synthesized faces $\mathbf{x}_{adv}$ via a GAN loss:

$$\mathcal{L}_{GAN} = \mathbb{E}_{\mathbf{x}} \left[\log Dsc(\mathbf{x})\right] + \mathbb{E}_{\mathbf{x}}[\log(1 - Dsc(\mathbf{x}_{adv}))]. \tag{3.4.5}$$

3.4.2.2 Adversarial Detector

Similar to prevailing adversarial detectors, the proposed detector also learns a decision boundary between real and adversarial images [166, 167, 179, 191]. A key difference, however, is that instead of utilizing pre-computed adversarial images from known attacks (*e.g.* FGSM and PGD) for training, the proposed detector learns to distinguish between real images and the *synthesized* set of diverse adversarial attacks output by the proposed adversarial generator in a self-supervised manner. This leads to the following advantage: *our proposed framework does not require a large collection of pre-computed adversarial face images for training*.

We utilize a binary CNN for distinguishing between real input probes, $\mathbf{x}$, and synthesized adversarial samples, $\mathbf{x}_{adv}$. The detector is trained with the Binary Cross-Entropy loss:

$$\mathcal{L}_{BCE} = \mathbb{E}_{\mathbf{x}} \left[\log \mathcal{D}(\mathbf{x})\right] + \mathbb{E}_{\mathbf{x}} \left[\log \left(1 - \mathcal{D}(\mathbf{x}_{adv})\right)\right]. \tag{3.4.6}$$

3.4.2.3 Adversarial Purifier

The objective of the adversarial purifier is to recover the real face image $\mathbf{x}$ given an adversarial face $\mathbf{x}_{adv}$. We aim to automatically remove the adversarial perturbations by training a neural network $\mathcal{P}ur$, referred as an adversarial purifier.

The adversarial purification process can be viewed as an inverted procedure of adversarial image synthesis. Contrary to the obfuscation loss in the adversarial generator, we require that the purified image, $\mathbf{x}_{pur}$, successfully matches to the subject in the input probe $\mathbf{x}$. Note that this

Attacks	TAR (%) @ 0.1% FAR($\downarrow$)	SSIM($\uparrow$)
FGSM [34]	26.23	0.83 ± 0.24
PGD [35]	04.91	0.89 ± 0.12
DeepFool [141]	36.18	0.91 ± 0.09
AdvFaces [16]	00.17	0.89 ± 0.02
GFLM [32]	68.03	0.55 ± 0.14
SemanticAdv [186]	70.05	0.71 ± 0.21
No Attack	99.82	1.00 ± 0.00

Table 3.4 Face recognition performance of ArcFace [9] under adversarial attack and average structural similarities (SSIM) between probe and adversarial images for obfuscation attacks on $485K$ genuine pairs in LFW [8].

can be achieved via a *feature recovery loss*, which is the opposite to the obfuscation loss, *i.e.*, $\mathcal{L}_{fr} = -\mathcal{L}_{obf}$.

Note that an adversarial face image, $\mathbf{x}_{adv} = \mathbf{x} + \delta$, is metrically close to the real image, $\mathbf{x}$, in the input space. If we can estimate δ, then we can retrieve the real face image. Here, the perturbations can be predicted by a neural network, $\mathcal{P}ur$. In other words, retrieving the purified image, $\mathbf{x}_{pur}$ involves: (1) subtracting the perturbations from the adversarial image, $\mathbf{x}_{pur} = \mathbf{x}_{adv} - \mathcal{P}ur(\mathbf{x}_{adv})$ and (2) ensuring that the *purification mask*, $\mathcal{P}ur(\mathbf{x}_{adv})$, is small so that we do not alter the content of the face image by a large magnitude. Therefore, we propose a hybrid perceptual loss that (1) ensures $\mathbf{x}_{pur}$ is as close as possible to the real image, $\mathbf{x}$ via a ℓ_1 reconstruction loss and (2) a loss that minimizes the amount of alteration, $\mathcal{P}ur(\mathbf{x}_{adv})$:

$$\mathcal{L}_{perc} = \mathbb{E}_{\mathbf{x}} \left|\left|\mathbf{x}_{pur} - \mathbf{x}\right|\right|_1 + \left|\left|\mathcal{P}ur(\mathbf{x}_{adv})\right|\right|_2 . \tag{3.4.7}$$

Finally, we also incorporate our detector to guide the training of our purifier. Note that, due to the diversity in synthesized adversarial faces, the proposed detector learns a tight decision boundary around real faces. This can serve as a strong self-supervisory signal to the purifier for ensuring that the purified images belong to the real face distribution. Therefore, we also incorporate the detector as a discriminator for the purifier via the proposed bonafide loss:

$$\mathcal{L}_{bf} = \mathbb{E}_{\mathbf{x}} \left[\log \mathcal{D}(\mathbf{x}_{pur})\right] . \tag{3.4.8}$$

	Detection Accuracy (%)	Year	FGS [34]	PGD [35]	DpFl. [141]	AdvF. [16]	GFLM [32]	Sem. [186]	Mean ± Std.
General	Gong *et al.* [166]	2017	98.94	97.91	95.87	92.69	**99.92**	**99.92**	97.54 ± 02.82
	ODIN [175]	2018	83.12	84.39	71.74	50.01	87.25	85.68	77.03 ± 14.34
	Steganalysis [178]	2019	88.76	89.34	75.97	54.30	58.99	78.62	74.33 ± 14.77
Face	UAP-D [167]	2018	61.32	74.33	56.78	51.11	65.33	76.78	64.28 ± 09.97
	SmartBox [172]	2018	58.79	62.53	51.32	54.87	50.97	62.14	56.77 ± 05.16
	Goswami *et al.* [176]	2019	84.56	91.32	89.75	76.51	52.97	81.12	79.37 ± 14.04
	Massoli *et al.* [179] (MLP)	2020	63.58	76.28	81.78	88.38	51.97	52.98	69.16 ± 15.29
	Massoli *et al.* [179] (LSTM)	2020	71.53	76.43	88.32	75.43	53.76	55.22	70.11 ± 13.35
	Agarwal *et al.* [182]	2020	94.44	95.38	91.19	74.32	51.68	87.03	87.03 ± 16.86
	Proposed FaceGuard	2021	**99.85**	**99.85**	**99.85**	**99.84**	99.61	99.85	**99.81 ± 00.10**

Table 3.5 Detection accuracy of SOTA adversarial face detectors in classifying six adversarial attacks synthesized for the LFW dataset [8]. Detection threshold is set as 0.5 for all methods. All baseline methods require training on pre-computed adversarial attacks on CASIA-WebFace [10]. On the other hand, the proposed *FaceGuard* is self-guided and generates adversarial attacks on the fly. Hence, it can be regarded as a *black-box* defense system.

3.4.2.4 Training Framework

We train the entire *FaceGuard* framework in Fig. 3.20 in an end-to-end manner with the following objectives:

$$\min_{\mathcal{G}} \mathcal{L}_{\mathcal{G}} = \mathcal{L}_{GAN} + \lambda_{obf} \cdot \mathcal{L}_{obf} + \lambda_{pt} \cdot \mathcal{L}_{pt} - \lambda_{div} \cdot \mathcal{L}_{div},$$

$$\min_{\mathcal{D}} \mathcal{L}_{\mathcal{D}} = \mathcal{L}_{BCE},$$

$$\min_{\mathcal{P}ur} \mathcal{L}_{\mathcal{P}ur} = \lambda_{fr} \cdot \mathcal{L}_{fr} + \lambda_{perc} \cdot \mathcal{L}_{perc} + \lambda_{bf} \cdot \mathcal{L}_{bf}.$$

At each training iteration, the generator attempts to fool the discriminator by synthesizing visually realistic adversarial faces while the discriminator learns to distinguish between real and synthesized images. On the other hand, in the same iteration, an external critic network, namely detector $\mathcal{D}$, learns a decision boundary between real and synthesized adversarial samples. Concurrently, the purifier $\mathcal{P}ur$ learns to invert the adversarial synthesis process. Note that there is a key difference between the discriminator and the detector: the generator is designed to specifically *fool* the discriminator but not necessarily the detector. We will show in our experiments that this crucial step prevents the detector from predicting $\mathcal{D}(\mathbf{x}) = 0.5$ for all $\mathbf{x}$ (see Tab. 3.7).

3.4.3 Experimental Settings

Datasets. We train *FaceGuard* on real face images in CASIA-WebFace [10] dataset and then evaluate on real and adversarial faces synthesized for LFW [8], Celeb-A [17] and FFHQ [18] datasets. CASIA-WebFace [10] comprises of $494,414$ face images from $10,575$[15] different subjects. LFW [8] contains $13,233$ face images of $5,749$ subjects. Since we evaluate defenses under obfuscation attacks, we consider subjects with at least two face images[16]. After this filtering, $9,164$ face images of $1,680$ subjects in LFW are available for evaluation. For brevity, experiments on CelebA and FFHQ are provided in Addendum (Sec. 3.4.6).

Implementation. The adversarial generator and purifier employ a convolutional encoder-decoder. The latent variable $\mathbf{z}$, a 128-dimensional feature vector, is fed as input to the generator through spatial padding and concatenation. The adversarial detector, a 4-layer binary CNN, is trained jointly with the generator and purifier. Empirically, we set $\lambda_{obf} = \lambda_{fr} = 10.0$, $\lambda_{pt} = \lambda_{perc} = 1.0$, $\lambda_{div} = 1.0$, $\lambda_{bf} = 1.0$ and $\epsilon = 3.0$. Training and network architecture details are provided in Addendum (Sec. 3.4.6).

Face Recognition Systems. In this study, we use two AFR systems: FaceNet [6] and ArcFace [9]. Recall that the proposed defense utilizes a face matcher, $\mathcal{F}$, for guiding the training process of the generator. However, the deployed AFR system may not be known to the defense system a priori. Therefore, unlike prevailing defense mechanisms [167, 172, 179], we evaluate the effectiveness of the proposed defense on an AFR system *different* from $\mathcal{F}$. We highlight the effectiveness of our proposed defense: *FaceGuard is trained on FaceNet, while the adversarial attack test set is designed to evade ArcFace.* Obfuscation attempts perturb real probes into adversarial ones. Ideally, deployed AFR systems (say, ArcFace), should be able to match a genuine pair comprised of an adversarial probe and a real enrolled face of the same subject. Therefore, regardless of real or adversarial probe, we assume that genuine pairs should *always* match as ground truth. Tab. 3.4 provides AFR performance of ArcFace under 6 SOTA adversarial attacks for $484,514$ genuine

[15] We removed 84 subjects in CASIA-WebFace that overlap with LFW.

[16] Obfuscation attempts only affect genuine pairs (two face images pertaining to the same subject).

pairs in LFW. It appears that some attacks, *e.g.*, AdvFaces [16], are effective in both low TAR and high SSIM, while some are less capable in both metrics.

3.4.4 Comparison with State-of-the-Art Defenses

In this section, we compare the proposed *FaceGuard* to prevailing defenses. We evaluate all methods via publicly available repositories provided by the authors (see Addendum (Sec. 3.4.6).). All baselines are trained on CASIA-WebFace [10].

SOTA Detectors. Our baselines include 9 SOTA detectors proposed both for general objects [166,175,178] and adversarial faces [167,172,176,179,182]. The detectors are trained on real and adversarial faces images synthesized via six adversarial generators for CASIA-WebFace [10]. Unlike all the baselines, *FaceGuard*'s detector does not utilize any pre-computed adversarial attack for training. We compute the classification accuracy for all methods on a dataset comprising of $9,164$ real images and $9,164$ adversarial face images per attack type in LFW.

In Tab. 3.5, we find that compared to the baselines, *FaceGuard* achieves the highest detection accuracy. Even when the 6 adversarial attack types are encountered in training, a binary CNN [166], still falls short compared to *FaceGuard*. This is likely because *FaceGuard* is trained on a diverse set of adversarial faces from the proposed generator. While the binary CNN has a small drop compared to FaceGuard in the seen attacks ($99.81\% \rightarrow 97.54\%$), it drops significantly on unseen adversarial attacks in testing.

Compared to hand-crafted features, such as PCA+SVM in UAP-D [167] and entropy detection in SmartBox [172], *FaceGuard* achieves superior detection results. Some baselines utilize AFR features for identifying adversarial inputs [176, 179]. We find that intermediate AFR features primarily represent the identity of the input face and do not appear to contain highly discriminative information for detecting adversarial faces.

Despite the robustness, *FaceGuard* misclassifies 28 out of $9,164$ real images in LFW [8] and falsely predicts 46 out of $54,984$ adversarial faces as real. From the latter, 44 are warped faces via GFLM [32] and the remaining two are synthesized via AdvFaces [16]. We find that *FaceGuard*

0.77 0.67 0.96 0.99

(a) Real faces falsely detected as adversarial

Real AdvFaces (0.42) Real AdvFaces (0.28)

(b) Adversarial faces falsely detected as real

Figure 3.21 Examples where the proposed *FaceGuard* fails to correctly detect (a) real faces and (b) adversarial faces. Detection scores $\in [0, 1]$ are given below each image, where 0 indicates real and 1 indicates adversarial face.

tends to misclassify real faces under extreme poses and adversarial faces that are occluded (*e.g.*, hats) (see Fig. 3.21).

Comparison with Adversarial Training & Purifiers. We also compare with prevailing defenses designing robust face matchers [11–13] and purifiers [14, 15, 184]. We conduct a verification experiment by considering all possible genuine pairs (two faces belonging to the same subject) in LFW [8]. For one probe in a genuine pair, we craft six different adversarial probes (one per attack type). In total, there are $484,514$ real pairs and $\sim 3M$ adversarial pairs. For a fixed match threshold[17], we compute the True Accept Rate (TAR) of successfully matching two images in a real or adversarial pair in Tab. 3.6. In other words, TAR is defined here as the ratio of genuine pairs above the match threshold.

ArcFace without any adversarial defense system achieves 34.27% TAR at 0.1% FAR under

[17]We compute the threshold at 0.1% FAR on all possible image pairs in LFW, *e.g.*, threshold @ 0.1% FAR for ArcFace is set at 0.36.

Defenses	**Year**	**Strategy** 485K pairs	**Real** $3M$ pairs	**Attacks**
No-Defense	–	-	99.82	34.27
Adv. Training [11]	2017	Robustness	96.42	11.23
Rob-GAN [12]	2019	Robustness	91.35	13.89
Feat. Denoising [164]	2019	Robustness	87.61	17.97
L2L [13]	2019	Robustness	96.89	16.76
MagNet [14]	2017	Purification	94.47	38.32
DefenseGAN [15]	2018	Purification	96.78	39.21
Feat. Distillation [183]	2019	Purification	94.64	41.77
NRP [184]	2020	Purification	97.54	61.44
A-VAE [185]	2020	Purification	93.71	51.99
Proposed FaceGuard	2021	Purification	**99.81**	**77.46**

Table 3.6 AFR performance (TAR (%) @ 0.1% FAR) of ArcFace under no defense and when ArcFace is trained via SOTA robustness techniques [11–13] or SOTA purifiers [14,15]. *FaceGuard* correctly passes majority of real faces to ArcFace and also purifies adversarial attacks.

attack. Adversarial training [11–13] inhibits the feature space of ArcFace, resulting in worse performance on both real and adversarial pairs. On the other hand, purification methods [14, 15, 184] can better retain face features in real pairs but their performance under attack is still undesirable.

Instead, the proposed *FaceGuard* defense system first detects whether an input face image is real or adversarial. If input faces are adversarial, they are further purified. From Tab. 3.6, we find that our defense system significantly outperforms SOTA baselines in protecting ArcFace [9] against attacks. Specifically, *FaceGuard*'s purifier enhances ArcFace's average TAR at 0.1% FAR under all six attacks (see Tab. 3.4) from $34.27\% \rightarrow 77.46\%$. In addition, *FaceGuard* also maintains similar face recognition performance on real faces (TAR on real pairs drop from $99.82\% \rightarrow 99.81\%$). Therefore, our proposed defense system ensures that benign users will not be incorrectly rejected while malicious attempts to evade the AFR system will be curbed.

3.4.5 Analysis of Our Approach

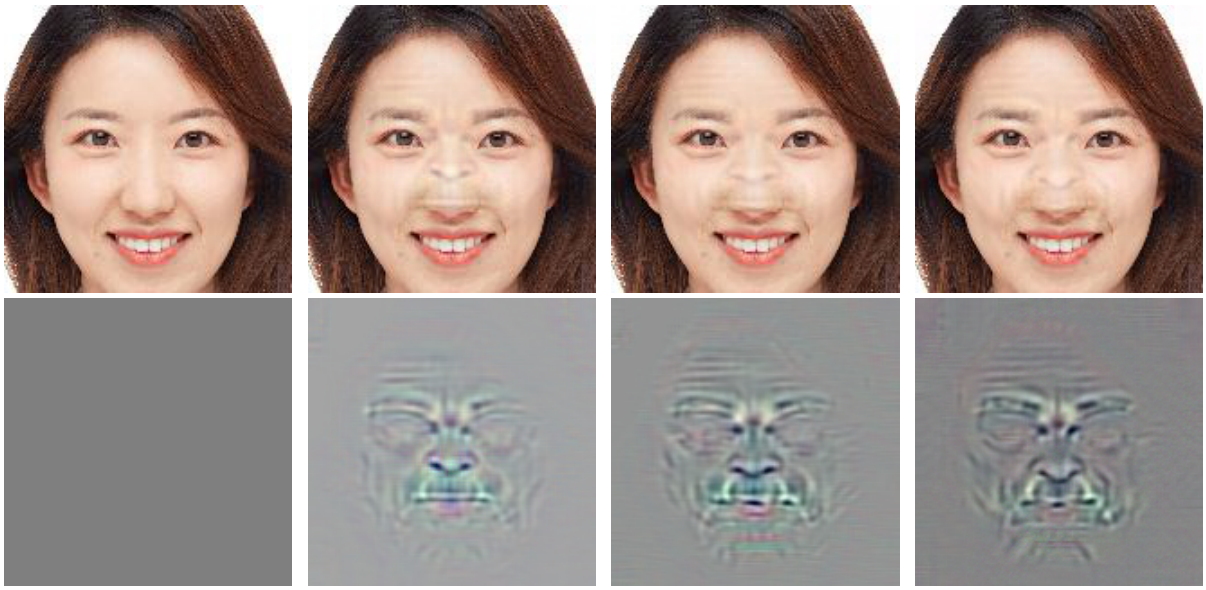
Quality of the Adversarial Generator. In Tab. 3.7, we see that without the proposed adversarial generator ("Without $\mathcal{G}$"), *i.e.*, a detector trained on the six known attack types, suffers from high

	Model	AdvFaces [16]	Mean $\pm$ Std.
Gen. $\mathcal{G}$	Without $\mathcal{G}$	91.72	97.12 ± 04.54
	Without $\mathcal{L}_{div}$	95.42	98.23 ± 01.33
	With $\mathcal{G}$ and $\mathcal{L}_{div}$	**99.84**	$\mathbf{99.81 \pm 00.10}$
Det. $\mathcal{D}$	$\mathcal{D}$ as Discriminator	50.00	75.25 ± 21.19
	$\mathcal{D}$ via Pre-Computed $\mathcal{G}$	52.01	69.37 ± 19.91
	$\mathcal{D}$ as Online Detector	**99.84**	$\mathbf{99.81 \pm 00.10}$

Table 3.7 Ablating training schemes of the generator $\mathcal{G}$ and detector $\mathcal{D}$. All models are trained on CASIA-WebFace [10]. *(Col. 3)* We compute the detection accuracy in classifying real faces in LFW [8] and the most challenging adversarial attack in Tab. 3.4, AdvFaces [16]. *(Col. 4)* The avg. and std. dev. of detection accuracy across all 6 adversarial attacks.

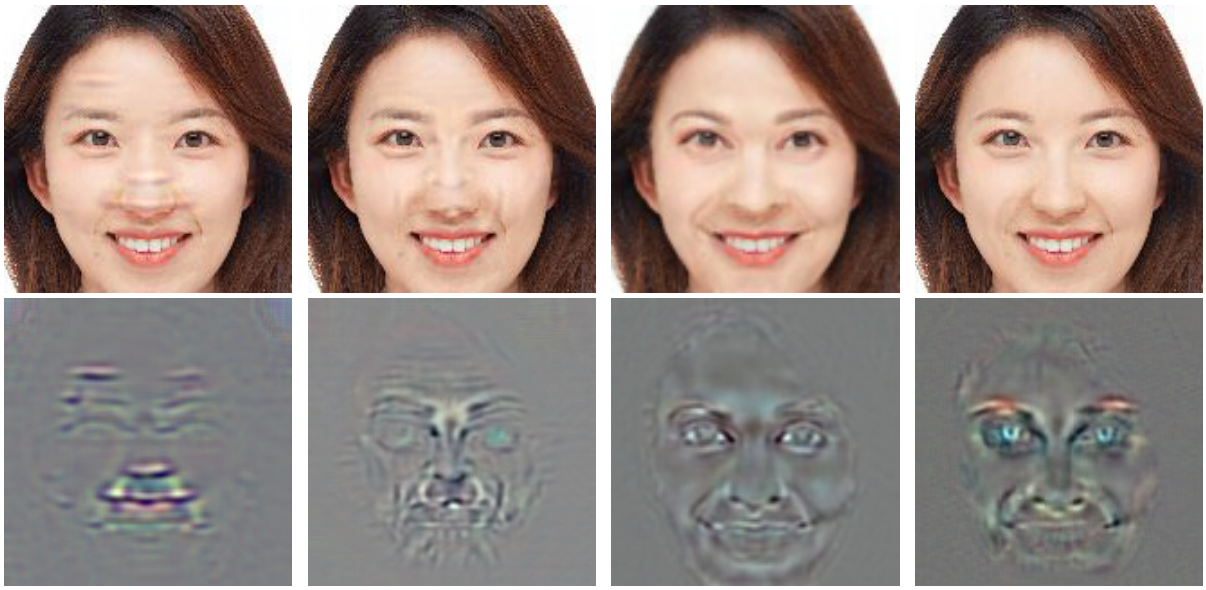
standard deviation. Instead, training a detector with a deterministic $\mathcal{G}$ ("Without $\mathcal{L}_{div}$"), leads to better generalization across attack types, since the detector still encounters variations in synthesized images as the generator learns to better generate adversarial faces. However, such a detector is still prone to overfitting to a few deterministic perturbations output by $\mathcal{G}$. Finally, *FaceGuard* with the diversity loss introduces diverse perturbations within and across training iterations (see Fig. 3.22).

Quality of the Adversarial Detector. The discriminator's task is similar to the detector; determine whether an input image is real or fake/adversarial. The key difference is that the generator is enforced to fool the discriminator, but not the detector. If we replace the discriminator with an adversarial detector, the generator continuously attempts to fool the detector by synthesizing images that are as close as possible to the real image distribution. By design, such a detector should converge to $Disc(\mathbf{x}) = 0.5$ for all $\mathbf{x}$ (real or adversarial). As we expect, in Tab. 3.7, we cannot rely on predictions made by such a detector ("$\mathcal{D}$ as Discriminator"). We try another variant: we first train the generator $\mathcal{G}$ and then train a detector to distinguish between real and pre-computed attacks via $\mathcal{G}$ ("$\mathcal{D}$ via Pre-Computed $\mathcal{G}$"). As we expect, the proposed methodology of training the detector in an online fashion by utilizing the synthesized adversarial samples output by $\mathcal{G}$ at any given iteration leads to a significantly robust detector ("$\mathcal{D}$ as Online Detector"). This can likely be attributed to the fact that a detector trained on-line encounters a much larger variation as the generator trains alongside. "$\mathcal{D}$ via Pre-Computed $\mathcal{G}$" is exposed only to within-iteration variations



Input Probe ($\mathbf{x}$) $\mathcal{G}(\mathbf{x}, \mathbf{z}_1)$ $\mathcal{G}(\mathbf{x}, \mathbf{z}_2)$ $\mathcal{G}(\mathbf{x}, \mathbf{z}_3)$

(a) Adversarial faces via random latents within the same iteration.



Iteration: $5K$ Iteration: $20K$ Iteration: $60K$ Iteration: $100K$

(b) Adversarial faces at different training iterations.

Figure 3.22 Adversarial faces synthesized by *FaceGuard* during training. Note the diversity in perturbations (a) within and (b) across iterations.

(from random latent sampling), however, '$\mathcal{D}$ as Online Detector" encounters variations *both* within and across training iterations (see Fig. 3.22).

Quality of the Adversarial Purifier. Recall that we enforced the purified image to be close to the

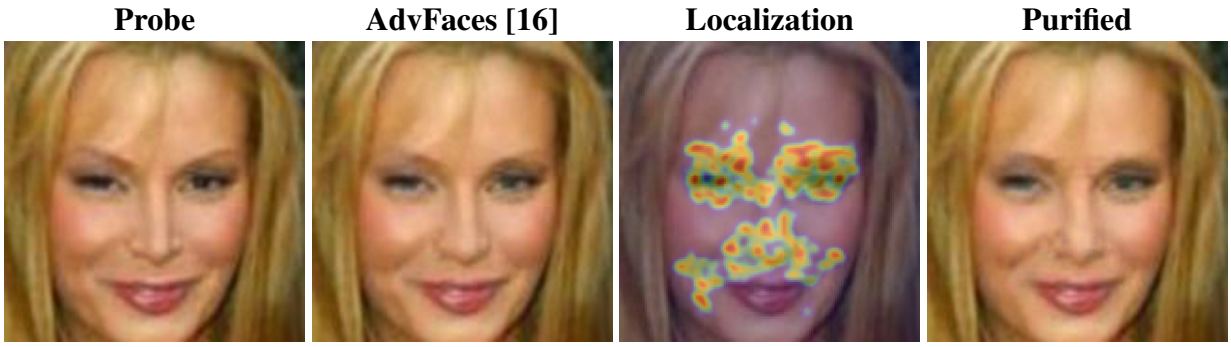
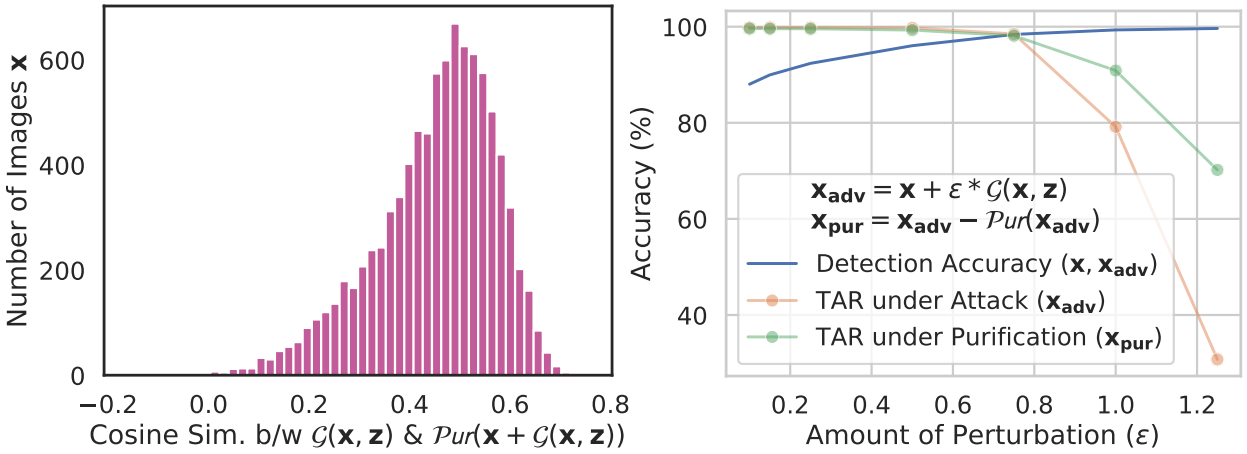


Figure 3.23 *FaceGuard* successfully purifies the adversarial image (red regions indicate adversarial perturbations localized by our purification mask). ArcFace [9] scores $\in [-1, 1]$ and SSIM $\in [0, 1]$ between an adversarial/purified probe and input probe are given below each image.



(a) (b)

Figure 3.24 (a) *FaceGuard*'s purification is correlated with its adversarial synthesis process. (b) Trade-off between detection and purification with respect to perturbation magnitudes. With minimal perturbation, detection is challenging while purifier maintains AFR performance. Excessive perturbations lead to easier detection with greater challenge in purification.

real face via a reconstruction loss. Thus, the purification and perturbation masks should be similar. In Fig. 3.24a, we shows that the two masks are indeed correlated by plotting the Cosine similarity distribution ($\in [-1, 1]$) between $\mathcal{G}(\mathbf{x}, \mathbf{z})$ and $\mathcal{P}ur(\mathbf{x} + \mathcal{G}(\mathbf{x}, \mathbf{z}))$ for all $9,164$ images in LFW.

Therefore, pixels in $\mathbf{x}_{adv}$ involved in the purification process should correspond to those that cause the image to be adversarial in the first place. Fig. 3.23 highlights that perturbed regions can be automatically localized via constructing a heatmap out of $\mathcal{P}ur(\mathbf{x}_{adv})$. In Fig. 3.24b, we investigate the change in AFR performance (TAR (%) @ 0.1% FAR) of ArcFace under attack (synthesized adversarial faces via $\mathcal{G}(\mathbf{x}, \mathbf{z})$) when the amount of perturbation is varied. We find that

(a) minimal perturbation is harder to detect but the purifier incurs minimal damage to the AFR, while, (b) excessive perturbations are easier to detect but increases the challenge in purification.

3.4.6 Addendum

3.4.6.1 Implementation Details

All the models in the chapter are implemented using Tensorflow r1.12. A single NVIDIA GeForce GTX 2080Ti GPU is used for training *FaceGuard* on CASIA-Webface [10] and evaluated on LFW [8], CelebA [17], and FFHQ [18].

3.4.6.2 Preprocessing

All face images are first passed through MTCNN face detector [41] to detect 5 facial landmarks (two eyes, nose and two mouth corners). Then, similarity transformation is used to normalize the face images based on the five landmarks. After transformation, the images are resized to 160×160. Before passing into *FaceGuard*, each pixel in the RGB image is normalized $\in [-1, 1]$ by subtracting 128 and dividing by 128. **All the testing images are from the identities in the test dataset.**

3.4.6.3 Network Architectures

The generator, $\mathcal{G}$ takes as input an real RGB face image, $\mathbf{x} \in \mathbb{R}^{160\times160\times3}$ and a 128-dimensional random latent vector, $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ and outputs a synthesized adversarial face $\mathbf{x}_{adv} \in \mathbb{R}^{160\times160\times3}$. Let `c7s1-k` be a 7×7 convolutional layer with k filters and stride 1. `dk` denotes a 4×4 convolutional layer with k filters and stride 2. `Rk` denotes a residual block that contains two 3×3 convolutional layers. `uk` denotes a $2\times$ upsampling layer followed by a 5×5 convolutional layer with k filters and stride 1. We apply Instance Normalization and Batch Normalization to the generator and discriminator, respectively. We use Leaky ReLU with slope 0.2 in the discriminator and ReLU activation in the generator. The architectures of the two modules are as follows:

- Generator:

  `c7s1-64,d128,d256,R256,R256,R256, u128, u64, c7s1-3`,

- Discriminator:

  `d32,d64,d128,d256,d512`.

A 1×1 convolutional layer with 3 filters and stride 1 is attached to the last convolutional layer of the discriminator for the patch-based GAN loss $\mathcal{L}_{GAN}$.

The purifier, $\mathcal{P}ur$, consists of the same network architecture as the generator:

- Purifier:

  `c7s1-64,d128,d256,R256,R256,R256, u128, u64, c7s1-3`.

We apply the `tanh` activation function on the last convolution layer of the generator and the purifier to ensure that the generated images are $\in [-1, 1]$. In the chapter, we denoted the output of the tanh layer of the generator as an "perturbation mask", $\mathcal{G}(\mathbf{x}, \mathbf{z}) \in [-1, 1]$ and $\mathbf{x} \in [-1, 1]$. Similarly, the output of the tanh layer of the purifier is referred to an "purification mask", $\mathcal{P}ur(\mathbf{x}_{adv}) \in [-1, 1]$ and $\mathbf{x}_{adv} \in [-1, 1]$. The final adversarial image is computed as $\mathbf{x}_{adv} = 2 \times \texttt{clamp}\left[\mathcal{G}(\mathbf{x}, \mathbf{z}) + \left(\frac{\mathbf{x}+1}{2}\right)\right]_0^1 - 1$. This ensures $\mathcal{G}(\mathbf{x}, \mathbf{z})$ can either add or subtract pixels from x when $\mathcal{G}(\mathbf{x}, \mathbf{z}) \neq 0$. When $\mathcal{G}(\mathbf{x}, \mathbf{z}) \to 0$, then $\mathbf{x}_{adv} \to \mathbf{x}$. Similarly, the final purified image is computed as $x_{pur} = 2 \times \texttt{clamp}\left[\left(\frac{\mathbf{x}_{adv}+1}{2}\right) - \mathcal{P}ur(\mathbf{x}_{adv})\right]_0^1 - 1$.

The external critic network, detector $\mathcal{D}$, comprises of a 4-layer binary CNN:

- Detector:

  `d32,d64,d128,d256,fc64,fc1`,

where `fcN` refers to a fully-connected layer with N neuron outputs.

3.4.6.4 Training Details

The generator, detector, and purifier are trained in an end-to-end manner via ADAM optimizer with hyperparameters $\beta_1 = 0.5$, $\beta_2 = 0.9$, learning rate of $1e-4$, and batch size 16. Algorithm 2 outlines the training algorithm.

Network Convergence. In Fig. 3.25, we plot the training loss across iterations when an adversar-

Algorithm 2 Training *FaceGuard*. All experiments in this work use $\alpha = 0.0001$, $\beta_1 = 0.5$, $\beta_2 = 0.9$, $\lambda_{obf} = \lambda_{fr} = 10.0$, $\lambda_{pt} = \lambda_{perc} = \lambda_{div} = 1.0$, $\epsilon = 3.0$, $m = 16$. For brevity, lg refers to log operation.

Input
  $\mathcal{X}$ Training Dataset
  $\mathcal{F}$ Cosine similarity by AFR
  $\mathcal{G}$ Generator with weights $\mathcal{G}_\theta$
  Dc Discriminator with weights Dc_θ
  $\mathcal{D}$ Detector with weights $\mathcal{D}_\theta$
  $\mathcal{P}ur$ Purifier with weights $\mathcal{P}ur_\theta$
  m Batch size
  α Learning rate
for number of training iterations **do**
  Sample a batch of probes $\{x^{(i)}\}_{i=1}^{m} \sim \mathcal{X}$
  Sample a batch of random latents $\{z^{(i)}\}_{i=1}^{m} \sim \mathcal{N}(0, I)$
  $\delta_{\mathcal{G}}^{(i)} = \mathcal{G}((x^{(i)}, z^{(i)})$
  $x_{adv}^{(i)} = x^{(i)} + \delta_{\mathcal{G}}^{(i)}$
  $\delta_{\mathcal{P}ur}^{(i)} = \mathcal{G}((x^{(i)}, z^{(i)})$
  $x_{pur}^{(i)} = x_{adv}^{(i)} - \delta_{\mathcal{P}ur}^{(i)}$

  $\mathcal{L}_{pt}^{\mathcal{G}} = \frac{1}{m}\left[\sum_{i=1}^{m} \max\left(\epsilon, ||\delta^{(i)}||_2\right)\right]$
  $\mathcal{L}_{obf}^{\mathcal{G}} = \frac{1}{m}\left[\sum_{i=1}^{m} \mathcal{F}\left(x^{(i)}, x_{adv}^{(i)}\right)\right]$
  $\mathcal{L}_{div}^{\mathcal{G}} = -\frac{1}{m}\left[\sum_{i=1}^{m}\left[\frac{\left|\left|\mathcal{G}(\mathbf{x},\mathbf{z}_1)^{(i)} - \mathcal{G}(\mathbf{x},\mathbf{z}_2)^{(i)}\right|\right|_1}{||\mathbf{z}_1 - \mathbf{z}_2||_1}\right]\right]$
  $\mathcal{L}_{GAN}^{\mathcal{G}} = \frac{1}{m}\left[\sum_{i=1}^{m} lg\left(1 - Dc(x_{adv}^{(i)})\right)\right]$
  $\mathcal{L}_{\mathcal{D}} = \frac{1}{m}\sum_{i=1}^{m}\left[lg\mathcal{D}(\mathbf{x}^{(i)}) + lg\left(1 - \mathcal{D}(\mathbf{x}_{adv}^{(i)})\right)\right]$
  $\mathcal{L}_{Dc} = \frac{1}{m}\sum_{i=1}^{m}\left[lg\left(Dc(x^{(i)})\right) + lg\left(1 - Dc(x_{adv}^{(i)})\right)\right]$
  $\mathcal{L}_{perc}^{\mathcal{P}ur} = \frac{1}{m}\sum_{i=1}^{m}\left[||x_{pur} - x||_1 + ||\mathcal{P}ur(x_{adv}^{(i)})||_1\right]$
  $\mathcal{L}_{fr}^{\mathcal{P}ur} = -\frac{1}{m}\left[\sum_{i=1}^{m} \mathcal{F}\left(x^{(i)}, x_{pur}\right)\right]$
  $\mathcal{L}_{bf}^{\mathcal{P}ur} = \frac{1}{m}\left[\sum_{i=1}^{m} lg\left(1 - \mathcal{D}(x_{pur})\right)\right]$
  $\mathcal{L}_{\mathcal{G}} = \mathcal{L}_{GAN}^{\mathcal{G}} + \lambda_{obf}\mathcal{L}_{obf} + \lambda_{pt}\mathcal{L}_{pt} + \lambda_{div}\mathcal{L}_{div}$
  $\mathcal{L}_{\mathcal{P}ur} = \lambda_{fr}\mathcal{L}_{fr} + \lambda_{perc}\mathcal{L}_{perc} + \lambda_{bf}\mathcal{L}_{bf}$
  $\mathcal{G}_\theta = \texttt{Adam}(\nabla_{\mathcal{G}}\mathcal{L}^{\mathcal{G}}, \mathcal{G}_\theta, \alpha, \beta_1, \beta_2)$
  $Dc\theta = \texttt{Adam}(\nabla_{Dc}\mathcal{L}^{Dc}, Dc_\theta, \alpha, \beta_1, \beta_2)$
  $\mathcal{D}_\theta = \texttt{Adam}(\nabla_{\mathcal{D}}\mathcal{L}^{\mathcal{D}}, \mathcal{D}_\theta, \alpha, \beta_1, \beta_2)$
  $\mathcal{P}ur_\theta = \texttt{Adam}(\nabla_{\mathcal{P}ur}\mathcal{L}^{\mathcal{P}ur}, \mathcal{P}ur_\theta, \alpha, \beta_1, \beta_2)$
end for

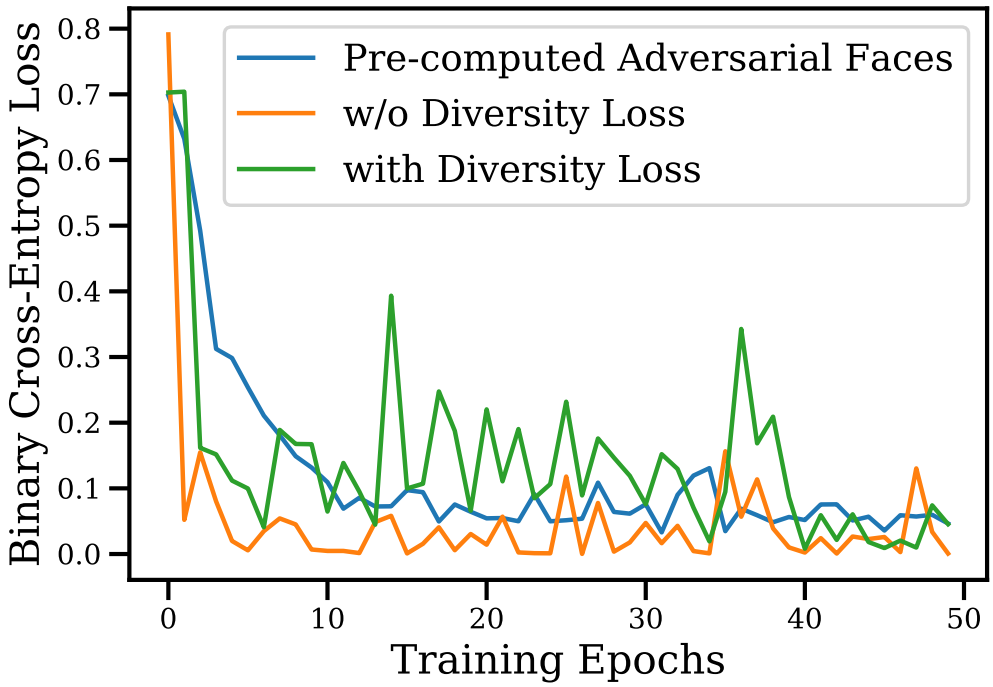


Figure 3.25 Training loss across iterations when an adversarial detection network is trained via pre-computed adversarial faces (blue), the proposed adv. generator but without the diversity (orange), and with the proposed diversity loss (green). The diversity loss prevents the network from overfitting to adversarial perturbations encountered during training.

ial detector is trained via pre-computed adversarial faces. In this case, the training loss converges to a low value and remains consistent across the remaining epochs. Such a detector may overfit to the fixed set of adversarial perturbations encountered in training. Instead of utilizing the pre-computed adversarial attacks, utilizing an adversarial generator in training (without $\mathcal{L}_{div}$), introduces challenging training samples.

FaceGuard with the diversity loss introduces diverse perturbations within a training iteration (see Fig. 3.22). In Fig. 3.25, we also observe that the training loss significantly fluctuates (epochs $8-40$) until convergence (epochs $40-50$). This indicates that throughout the training (within and across training iterations), the proposed generator synthesizes strong and diverse range of adversarial faces that continuously regularizes the training of the adversarial detector.

	Detection Accuracy (%)	Year	LFW [34]	CelebA [17]	FFHQ [18]
General	Gong *et al.* [166]	2017	97.54 ± 02.82	94.38 ± 04.48	96.89 ± 02.07
	ODIN [175]	2018	77.03 ± 14.34	68.95 ± 19.64	74.63 ± 08.16
	Steganalysis [178]	2019	74.33 ± 14.77	72.53 ± 11.30	71.09 ± 09.86
Face	UAP-D [167]	2018	64.28 ± 09.97	63.19 ± 16.49	68.65 ± 08.73
	SmartBox [172]	2018	56.77 ± 05.16	54.85 ± 09.33	57.19 ± 09.55
	Goswami *et al.* [176]	2019	79.37 ± 14.04	74.70 ± 13.88	80.03 ± 09.24
	Massoli *et al.* [179] (MLP)	2020	69.16 ± 15.29	61.78 ± 11.34	66.26 ± 10.06
	Massoli *et al.* [179] (LSTM)	2020	70.11 ± 13.35	63.67 ± 16.21	69.58 ± 07.91
	Agarwal *et al.* [182]	2020	87.03 ± 16.86	85.81 ± 15.64	86.70 ± 11.04
	Proposed FaceGuard	2021	$\mathbf{99.81 \pm 00.10}$	$\mathbf{98.73 \pm 00.92}$	$\mathbf{99.35 \pm 00.09}$

Table 3.8 Average and standard deviation of detection accuracies of SOTA adversarial face detectors in classifying six adversarial attacks synthesized for the LFW [8], CelebA [17], and FFHQ [18] datasets. Detection threshold is set as 0.5 for all methods. All baseline methods require training on pre-computed adversarial attacks on CASIA-WebFace [10]. On the other hand, the proposed *FaceGuard* is self-guided and generates adversarial attacks on the fly. Hence, it can be regarded as a *black-box* defense system.

3.4.6.5 Baselines

We evaluate all defense methods via publicly available repositories provided by the authors. Only modification made is to replace their training datasets with CASIA-WebFace [10]. We provide the public links to the author codes below:

- Gong *et al.* [166]: https://github.com/gongzhitaao/adversarial-classifier
- UAP-D [167]/SmartBox *et al.* [172]: https://github.com/akhil15126/SmartBox
- Massoli *et al.* [179]: https://github.com/fvmassoli/trj-based-adversarials-detection
- Adversarial Training [11]: https://github.com/locuslab/fast_adversarial
- Rob-GAN [12]: https://github.com/xuanqing94/RobGAN
- L2L [13]: https://github.com/YunseokJANG/l2l-da
- MagNet [14]: https://github.com/Trevillie/MagNet
- DefenseGAN [15]: https://github.com/kabkabm/defensegan
- NRP [184]: https://github.com/Muzammal-Naseer/NRP

Attacks are also synthesized via publicly available author codes:

	Known			Unseen		
	FGSM [34]	**PGD [35]**	**DeepFool [141]**	**AdvFaces [16]**	**GFLM [32]**	**SemanticAdv [186]**
Gong *et al.* [166]	94.51	92.21	94.12	68.63	50.00	50.21
UAP-D [167]	63.65	69.33	56.38	60.81	50.12	50.28
SmartBox [172]	58.79	62.53	51.32	54.87	50.97	62.14
Massoli *et al.* [179] (MLP)	78.35	82.52	91.21	55.57	50.00	50.00
Massoli *et al.* [179] (LSTM)	74.61	86.43	94.73	62.43	50.00	50.00

(a)

	Known			Unseen		
	AdvFaces [16]	**GFLM [32]**	**SemanticAdv [186]**	**FGSM [34]**	**PGD [35]**	**DeepFool [141]**
Gong *et al.* [166]	81.39	96.72	98.97	84.46	57.00	72.32
UAP-D [167]	68.78	54.31	77.46	51.64	50.32	52.01
SmartBox [172]	54.87	50.97	62.14	58.79	62.53	51.32
Massoli *et al.* [179] (MLP)	77.64	86.54	94.78	55.20	51.32	52.90
Massoli *et al.* [179] (LSTM)	81.42	92.62	96.76	52.74	65.43	54.84

(b)

	Known					
	FGSM [34]	**PGD [35]**	**DeepFool [141]**	**AdvFaces [16]**	**GFLM [32]**	**SemanticAdv [186]**
Gong *et al.* [166]	98.94	97.91	95.87	92.69	**99.92**	**99.92**
UAP-D [167]	61.32	74.33	56.78	51.11	65.33	76.78
SmartBox [172]	58.79	62.53	51.32	54.87	50.97	62.14
Massoli *et al.* [179] (MLP)	63.58	76.28	81.78	88.38	51.97	52.98
Massoli *et al.* [179] (LSTM)	71.53	76.43	88.32	75.43	53.76	55.22
	Unseen					
Proposed FaceGuard	**99.85**	**99.85**	**99.85**	**99.84**	99.61	99.85

(c)

Table 3.9 Detection accuracy of SOTA adversarial face detectors in classifying six adversarial attacks synthesized for the LFW dataset [8] under various known and unseen attack scenarios. Detection threshold is set as 0.5 for all methods.

- FGSM/PGD/DeepFool: https://github.com/tensorflow/cleverhans
- AdvFaces: https://github.com/ronny3050/AdvFaces
- GFLM: https://github.com/alldbi/FLM
- SemanticAdv: https://github.com/AI-secure/SemanticAdv

3.4.6.6 Additional Datasets

In Tab. 3.8, we report average and standard deviation of detection rates of the proposed *FaceGuard* and other baselines on the 6 adversarial attacks synthesized on LFW [8], CelebA [17], and FFHQ [18] (following the same protocol as Tab. 3.5). For CelebA, we synthesize a total of $19,962 \times 6 = 119,772$ adversarial samples for $19,962$ real samples in the CelebA testing split [17]. We also synthesize $4,974 \times 6 = 29,844$ adversarial samples for $4,974$ real faces in FFHQ testing

split [18]. We find that the proposed *FaceGuard* outperforms all baselines in all three face datasets.

3.4.6.7 Overfitting in Prevailing Detectors

In Tab. 3.9, we provide the detection rates of prevailing SOTA detectors in detecting six adversarial attacks in LFW [8] when they are trained on different attack subsets. We highglight the overfitting issue when (a) SOTA detectors are trained on gradient-based adversarial attacks (FGSM [34], PGD [35], and DeepFool [141]) and tested on gradient-based and learning-based attacks (AdvFaces [16], GFLM [32], and SemanticAdv [186]), and (b) vice-versa. Tab. 3.9(c) reports the detection performance of SOTA detectors when all six attacks are available for training.

We find that detection accuracy of SOTA detectors significantly drops when tested on a subset of attacks not encountered during their training. Instead, the proposed *FaceGuard* maintains robust detection accuracy without even training on the pre-computed samples from any known attacks.

3.4.6.8 Qualitative Results

Generator Results. Fig. 3.26 shows examples of synthesized adversarial faces via the proposed adversarial generator $\mathcal{G}$. Note that the generator takes the input prob $\mathbf{x}$ and a random latent $\mathbf{z}$. We show synthesized perturbation masks and corresponding adversarial faces for three randomly sampled latents. We observe that the synthesized adversarial images evades ArcFace [9] while maintaining high structural similarity between adversarial and input probe.

Purifier Results. We show examples of purified images via *FaceGuard* and baselines including MagNet [14] and DefenseGAN [15] in Fig. 3.27. We observe that, compared to baselines, purified images synthesized via *FaceGuard* are visually realistic with minimal changes compared to the ground truth real probe. In addition, compared to the two baselines, *FaceGuard*'s purifier protects ArcFace [9] matcher from being evaded by the six adversarial attacks.

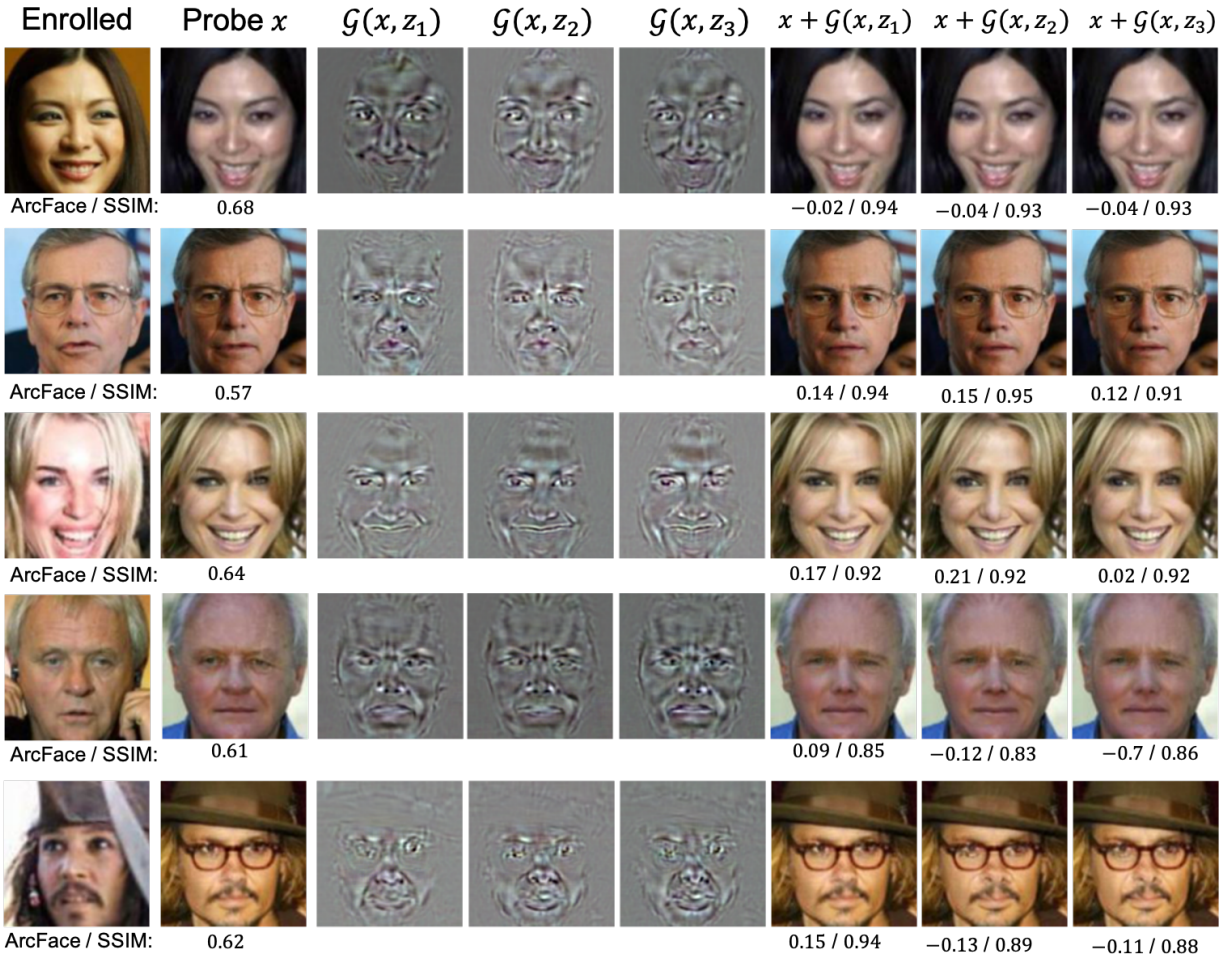


Figure 3.26 Examples of generated adversarial images along with corresponding perturbation masks obtained via *FaceGuard*'s generator $\mathcal{G}$ for three randomly sampled $\mathbf{z}$. Cosine similarity scores via ArcFace [9] $\in [-1, 1]$ and SSIM $\in [0, 1]$ between synthesized adversarial and input probe are given below each image. A score above **0.36** (threshold @ 0.1% False Accept Rate) indicates that two faces are of the same subject.

3.4.6.9 Additional Results on Purifier

Perturbation and Purification Masks. In the main text, we found that the perturbation and purification masks are correlated with an average Cosine similarity of 0.52. We show five pairs of perturbation and purification masks ranked by the Cosine similarity between them (highest to lowest). We observe that purification mask is better correlated when perturbations are more local. Slightly perturbing entire faces poses to be challenging for the proposed purifier.

Effect of Perturbation Amount. We also studied the effect of perturbation amount on detection

and purification results in the main text. We observed a trade-off between detection and purification with respect to perturbation magnitudes. With minimal perturbation, detection is challenging while purifier maintains AFR performance. Excessive perturbations lead to easier detection with greater challenge in purification. In Fig. 3.29, show examples of synthesized adversarial faces for different perturbation amounts and their corresponding purified images. We find that detection scores improve with larger perturbation. Aligned with our earlier findings, due to the proposed bonafide loss, $\mathcal{L}_{bf}$, purified faces are continuously detected as real by the detector which explains why the purifier maintains AFR performance with increasing perturbation amount.

Effect of Purification on ArcFace Embeddings. In order to investigate the effect of purification on a matcher's feature space, we extract face embeddings of real images, their corresponding adversarial images via the challenging AdvFaces [16] attack, and purified images, via the SOTA ArcFace matcher. In total, we extract feature vectors from $1,456$ face images of 10 subjects in the LFW dataset [8]. In Fig. 3.10, we plot the 2D t-SNE visualization of the face embeddings for the 10 subjects. The identity clusterings can be clearly observed from real, adversarial, and purified images. In particular, we observe that some adversarial faces pertaining to a subject moves farther from its identity cluster while the proposed purifier draws them back. Fig. 3.30 illustrates that the proposed purifier indeed enhances face recognition performance of ArcFace under attack from 34.27% TAR @ 0.1% FAR under no defense to 77.46% TAR @ 0.1% FAR.

3.5 Summary

This chapter first proposes a new method of adversarial face synthesis, namely *AdvFaces*, that automatically generates adversarial face images with imperceptible perturbations evading state-of-the-art face matchers. With the help of a GAN, and the proposed perturbation and identity losses, AdvFaces learns the set of pixel locations required by face matchers for identification and only perturbs those salient facial regions (such as eyebrows and nose). Once trained, AdvFaces generates high quality and perceptually realistic adversarial examples that are benign to the human

eye but can evade state-of-the-art black-box face matchers, while outperforming other state-of-the-art adversarial face methods.

With the introduction of sophisticated adversarial attacks on AFR systems, such as geometric warping and GAN-synthesized adversarial attacks, adversarial defense needs to be robust and generalizable. Without utilizing any pre-computed training samples from known adversarial attacks, the proposed *FaceGuard* achieved state-of-the-art detection performance against 6 different adversarial attacks. *FaceGuard*'s purifier also enhanced ArcFace's recognition performance under adversarial attacks.

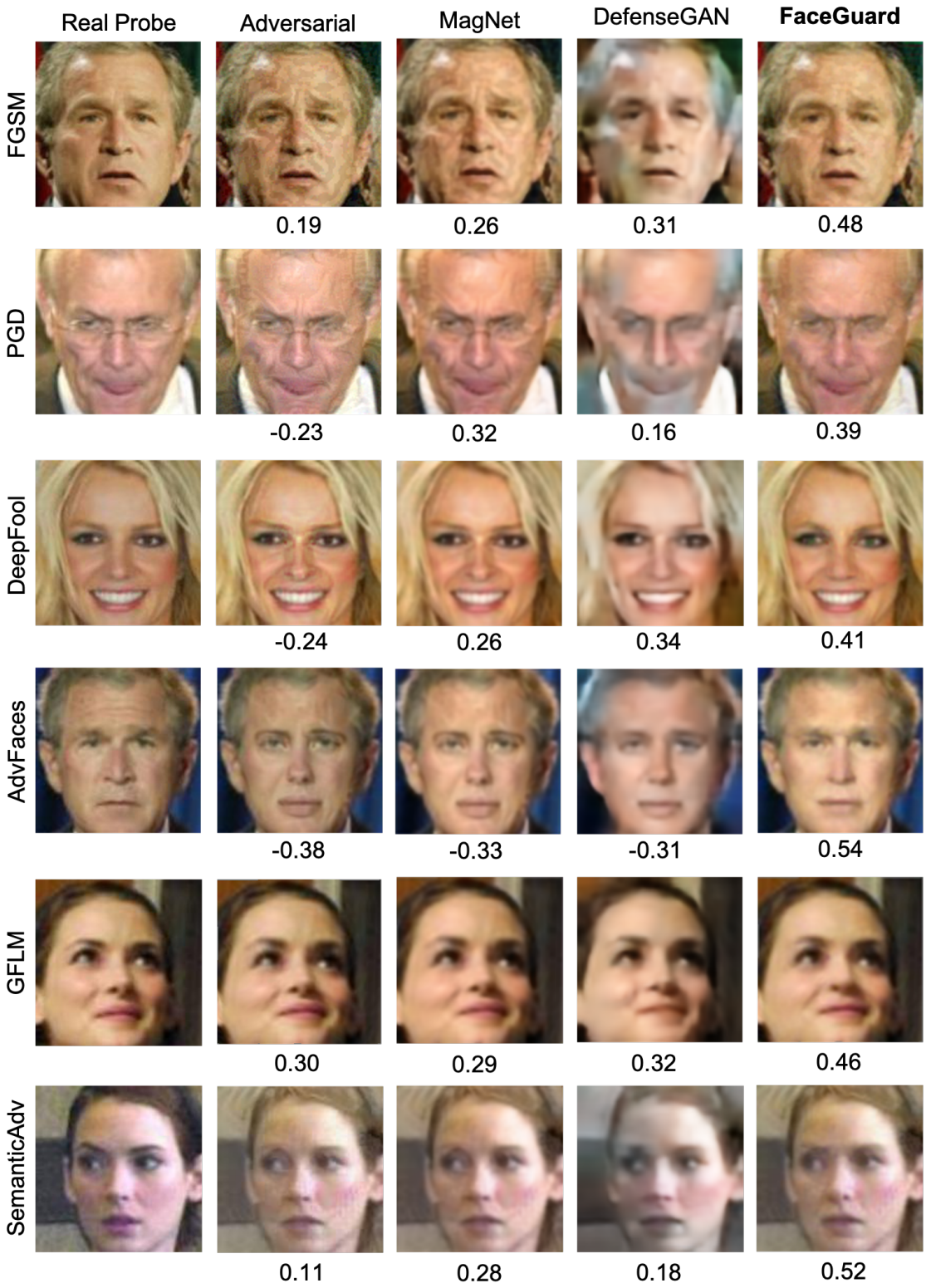


Figure 3.27 Examples of purified images via MagNet [14], DefenseGan [15], and proposed *FaceGuard* purifiers for six adversarial attacks. Cosine similarity scores via ArcFace [9] $\in [-1, 1]$ are given below each image. A score above **0.36** (threshold @ 0.1% False Accept Rate) indicates that two faces are of the same subject.

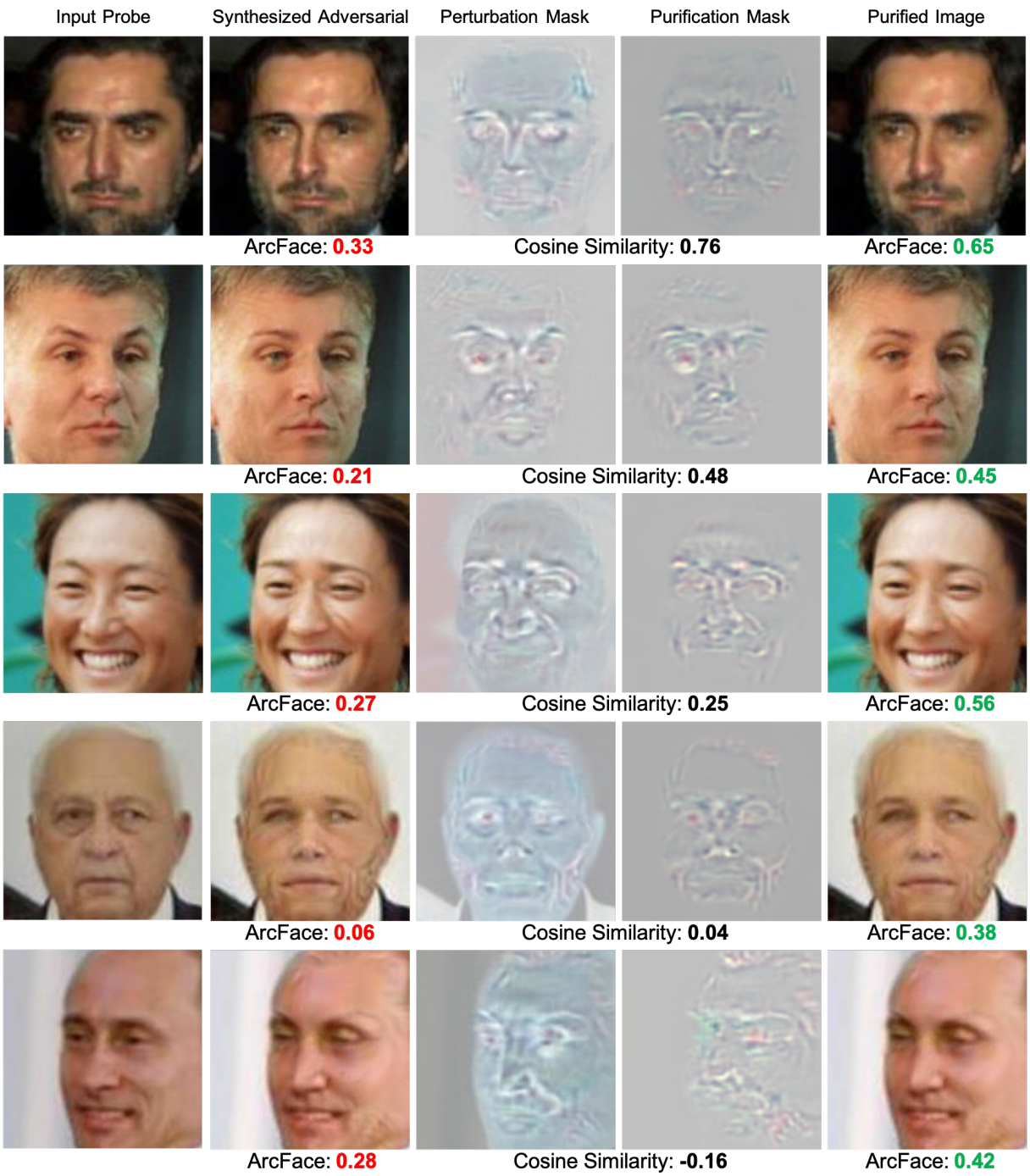


Figure 3.28 Examples of synthesized adversarial images via the proposed adversarial generator and corresponding purified images. Cosine similarity between perturbation and purification masks given below each row along with ArcFace scores between synthesized adversarial/purified image and real probe. A score above **0.36** (threshold @ 0.1% False Accept Rate) indicates that two faces are of the same subject. Even with lower correlation between perturbation and purification masks (rows 3-5), the purified images can still be identified as the correct identity. Notice that the purifier primarily alters the eye color, nose, and subdues adversarial perturbations in foreheads. Zoom in for details.

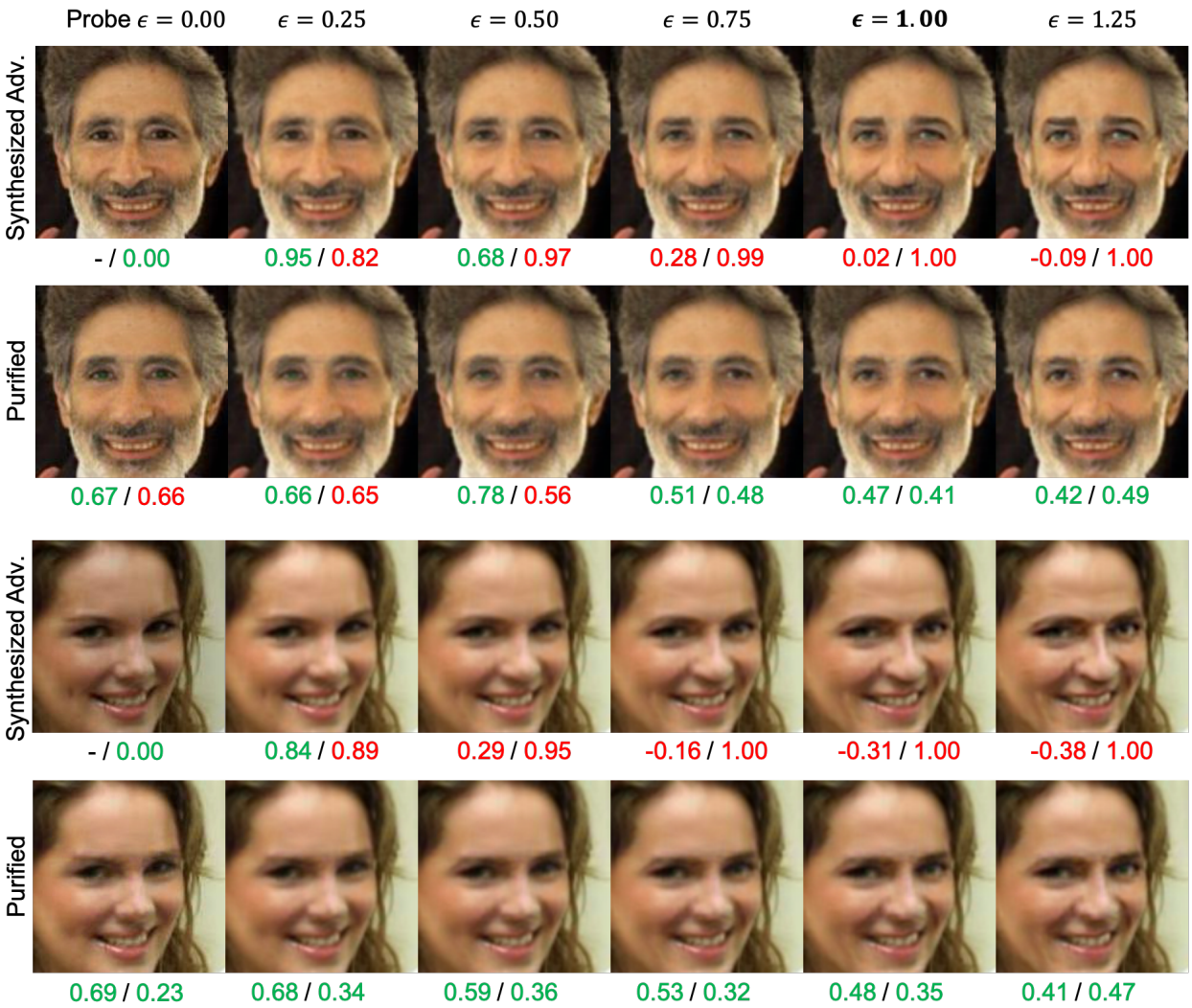


Figure 3.29 ArcFace $\in [-1, 1]$ / Detection scores $\in [0, 1]$ when perturbation amount is varied ($\epsilon = \{0.25, 0.50, 0.75, 1.00, 1.25\}$). Detection scores above 0.5 are predicted as adversarial images while ArcFace scores above **0.36** (threshold @ 0.1% False Accept Rate) indicate that two faces are of the same subject. *FaceGuard* is trained on $\epsilon = 1.00$. The detection scores improve as perturbation amount increases, whereas, majority of purified images are detected as real. Even when purified images fail to be classified as real by the detector, purification maintain high AFR performance.

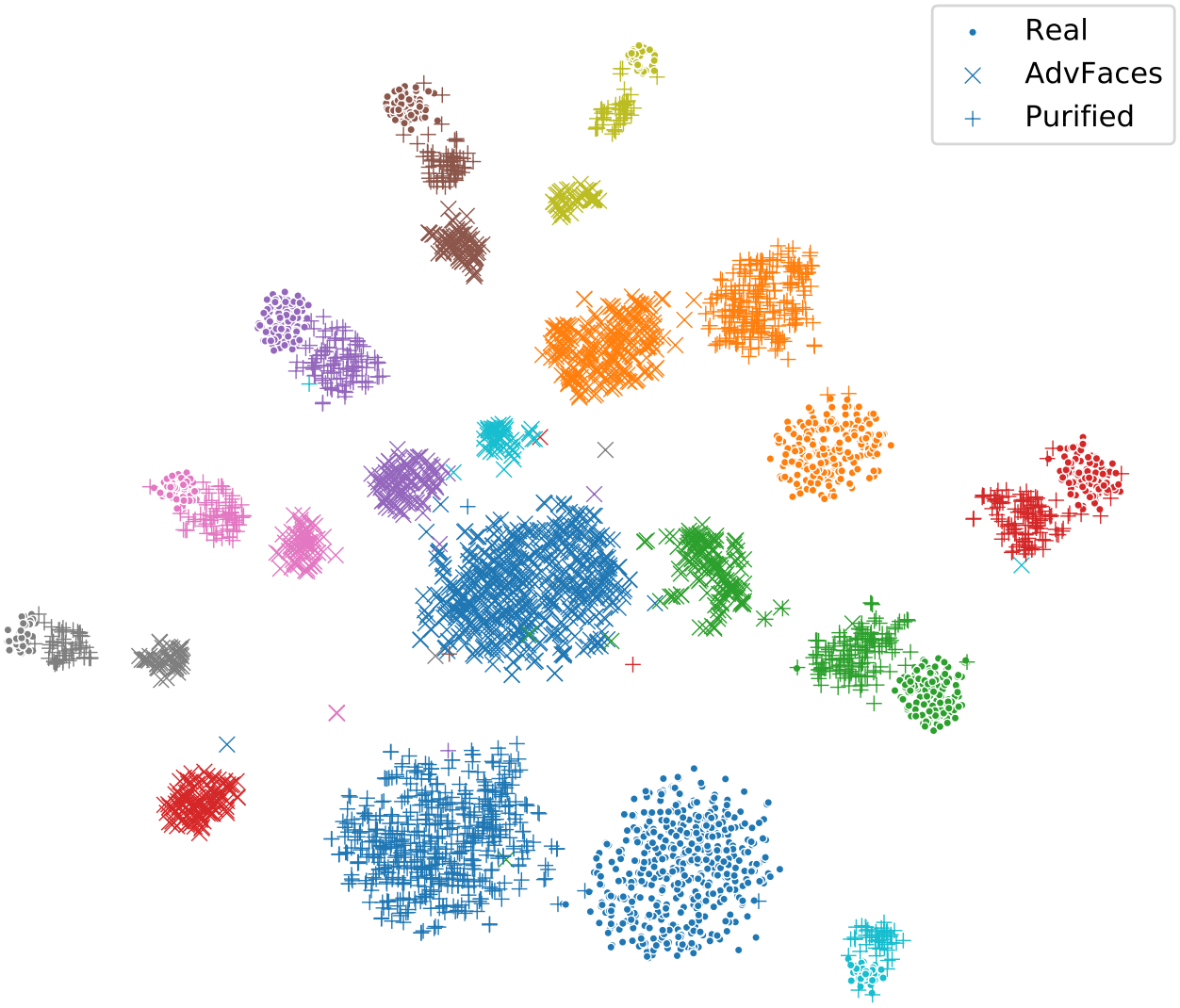


Figure 3.30 2D t-SNE visualization of face representations extracted via ArcFace from $1,456$ (a) real, (b) AdvFaces [16], and (c) purified images belonging to 10 subjects in LFW [8]. Example AdvFaces [16] pertaining to a subject moves farther from its identity cluster while the proposed purifier draws them back.

Chapter 4

Unified Detection of Digital and Physical Face Attacks

In the previous chapters, we proposed individual solutions to enhance the robustness of AFR system against physical and digital attacks. However, three broad categories of face attacks have been identified in literature, namely spoofs, digital manipulation, and adversarial faces. Since the exact type of face attack may not be known *a priori*, a generalizable detector that can defend an AFR system against any of the three attack categories is of utmost importance. In this chapter, our main focus is to design a universal face attack detection framework that can reliably detect attacks from all three categories.

4.1 Introduction

The foremost challenge facing AFR systems is their vulnerability to *face attacks*. For instance, an attacker can hide his identity by wearing a 3D mask [58], or intruders can assume a victim's identity by digitally swapping their face with the victim's face image [20]. With unrestricted access to the rapid proliferation of face images on social media platforms, launching attacks against AFR systems has become even more accessible. Given the growing dissemination of "fake news" and "deepfakes" [59], the research community and social media platforms alike are pushing towards

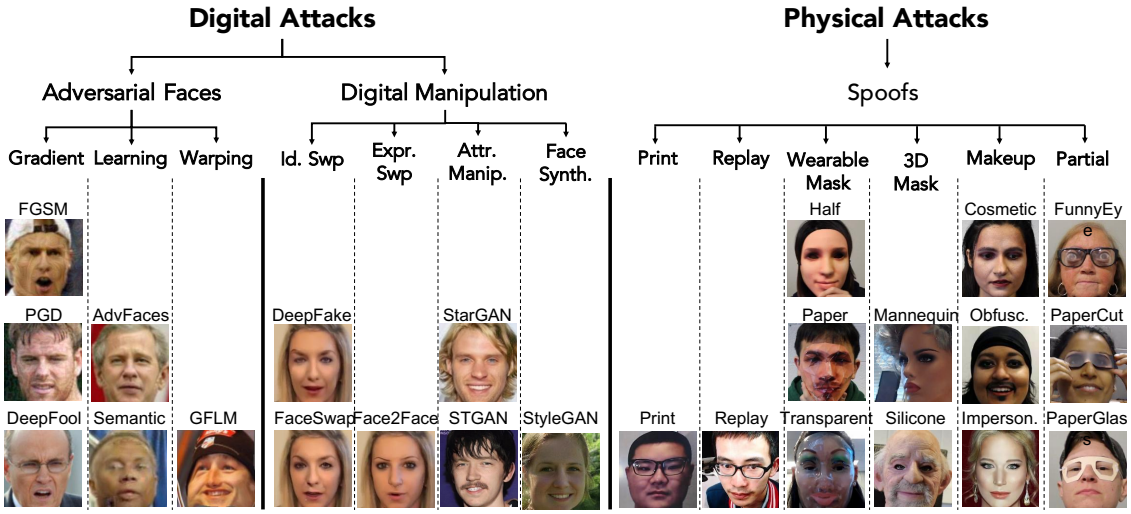


Figure 4.1 Face attacks against AFR systems are continuously evolving in both digital and physical spaces. Given the diversity of the face attacks, prevailing methods fall short in detecting attacks across all three categories (*i.e.*, adversarial, digital manipulation, and spoofs). This work is among the first to define the task of face attack detection on the 25 attack types across 3 categories shown here.

generalizable defense against continuously evolving and sophisticated face attacks.

In literature, face attacks can be broadly classified into three attack categories: (i) Spoof attacks: artifacts in the *physical* domain (*e.g.*, 3D masks, eye glasses, replaying videos) [1], (ii) Adversarial attacks: imperceptible noises added to probes for evading AFR systems [60], and (iii) Digital manipulation attacks: entirely or partially modified photo-realistic faces using generative models [20]. Within each of these categories, there are different attack types. For example, each spoof medium, *e.g.*, 3D mask and makeup, constitutes one attack type, and there are 13 common types of spoof attacks [1]. Likewise, in adversarial and digital manipulation attacks, each attack model, designed by unique objectives and losses, may be considered as one attack type. Thus, the attack categories and types form a 2-layer tree structure encompassing the diverse attacks (see Fig. 4.1). Such a tree will inevitably grow in the future.

In order to safeguard AFR systems against these attacks, numerous face attack detection approaches have been proposed [20, 21, 61–63]. Despite impressive detection rates, prevailing research efforts focus on a few attack types within *one* of the three attack categories. Since the exact type of face attack may not be known *a priori*, a generalizable detector that can defend an AFR

system against any of the three attack categories is of utmost importance.

Due to the vast diversity in attack characteristics, from glossy 2D printed photographs to imperceptible perturbations in adversarial faces, we find that learning a single *unified* network is inadequate. Even when prevailing state-of-the-art (SOTA) detectors are trained on all 25 attack types, they fail to generalize well during testing. Via ensemble training, we comprehensively evaluate the detection performance on fusing decisions from three SOTA detectors that individually excel at their respective attack categories. However, due to the diversity in attack characteristics, decisions made by each detector may not be complementary and result in poor detection performance across all 3 categories.

This research is among the first to focus on detecting *all* 25 *attack types* known in literature (6 adversarial, 6 digital manipulation, and 13 spoof attacks). Our approach consists of (i) automatically clustering attacks with similar characteristics into distinct groups, and (ii) a multi-task learning framework to learn salient features to distinguish between bona fides and coherent attack types, while early sharing layers learn a joint representation to distinguish bona fides from any generic attack.

This work makes the following contributions:

- Among the first to define the task of face attack detection on 25 attack types across 3 attack categories: adversarial faces, digital face manipulation, and spoofs.
- A novel **uni**fied **f**ace **a**ttack **d**etection framework, namely *UniFAD*, that automatically clusters similar attacks and employs a multi-task learning framework to detect digital and physical attacks.
- Proposed *UniFAD* achieves SOTA detection performance, TDR = 94.73% @ 0.2% FDR on a large fake face dataset, namely *GrandFake*. To the best of our knowledge, *GrandFake* is the largest face attack dataset studied in literature in terms of the number of diverse attack types.
- Proposed *UniFAD* allows for further identification of the attack categories, *i.e.*, whether attacks are adversarial, digitally manipulated, or contains physical spoofing artifacts, with a

	Study	Year	# BonaFides	# Attacks	# Types
Adversarial	UAP-D [167]	2018	$9,959$	$29,877$	1
	Goswami *et al.* [176]	2019	$16,685$	$50,055$	3
	Agarwal *et al.* [182]	2020	$24,042$	$72,126$	3
	Massoli *et al.* [179]	2020	$169,396$	$1M$	6
	FaceGuard [19]	2020	$507,647$	$3M$	6
Digital Manip.	Zhou *et al.* [214]	2018	$2,010$	$2,010$	2
	Yang *et al.* [215]	2018	$241(I)/49(V)$	$252(I)/49(V)$	1
	DeepFake [216]	2018	$-$	$620(V)$	1
	FaceForensics++ [216]	2019	$1,000(V)$	$3,000(V)$	3
	FakeSpotter [217]	2019	$6,000$	$5,000$	2
	DFFD [20]	2020	$58,703$	$240,336$	7
Phys. Spoofs	Replay-Attack [4]	2012	$200(V)$	$1,000(V)$	3
	MSU MFSD [85]	2015	$160(V)$	$280(V)$	3
	OuluNPU [2]	2017	$990(V)$	$3,960(V)$	4
	SiW [218]	2018	$1,320(V)$	$3,158(V)$	6
	SiW-M [1]	2019	$660(V)$	$960(V)$	13
	GrandFake (ours)	**2021**	$\mathbf{341,738}$	$\mathbf{447,674}$	$\mathbf{25}$

Table 4.1 Face attack datasets with no. of bona fide images, no. of attack images, and no. of attack types. Here, I denotes images and V refers to videos.

classification accuracy of 97.37%.

4.2 Related Work

Individual Attack Detection. Early work on face attack detection primarily focused on one or two attack types in their respective categories. Studies on adversarial face detection [166, 176] primarily involved detecting gradient-based attacks, such as FGSM [34], PGD [219], and DeepFool [141]. DeepFakes were among the first studied digital attack manipulation [214–216], however, generalizability of the proposed methods to a larger number of digital manipulation attack types is unsatisfactory [220]. Majority of face anti-spoofing methods focus on print and replay attacks [2, 66, 85, 89, 91, 91, 92, 94, 112, 123, 135, 218, 221, 222].

Over the years, a clear trend in the increase of attack types in each category can be observed in Tab. 4.1. Since a community of attackers dedicate their efforts to craft new attacks, it is imperative to comprehensively evaluate existing solutions against a large number of attack types.

Joint Attack Detection. Recent studies have used multiple attack types in order to defend against face attacks. For *e.g.*, FaceGuard [19] proposed a generalizable defense against 6 adversarial attack types. The Diverse Fake Face Dataset (DFFD) [20] includes 7 digital manipulation attack types. In the spoof attack category, recent studies focus on detecting 13 spoof types.

Majority of the works tackling multiple attack types pose the detection as a binary classification problem with a single network learning a joint feature space. For simplicity, we refer to such a network architecture as *JointCNN*. For instance, it is common in adversarial face detection to train a JointCNN with bona fide faces and adversarial attacks synthesized on-the-fly by a generative network [12, 13, 19, 184]. On the other hand, majority of the proposed defenses against digital manipulation, fine-tune a pre-trained JointCNN (*e.g.*, Xception [223]) on bona fide faces and all available digital manipulation attacks [20, 77, 217]. Due to the availability of a wide variety of physical spoof artifacts in face anti-spoofing datasets (*e.g.*, eyeglasses, print and replay instruments, masks, *etc*) along with evident cues for detecting them, studies on anti-spoofs are more sophisticated. The associated JointCNN employs either auxiliary cues, such as depth map and heart pulse signals (rPPG) [88, 127, 218], or a "compactness" loss to prevent overfitting [61, 224]. Recently Stehouwer *et al.* [225] attempt to learn a spoof detector from imagery of generic objects and apply it to face anti-spoofing. While jointly detecting multiple attack types is promising, detecting attack types *across* different categories is of the utmost importance. An early attempt proposed a defense against 4 attack types (3 spoofs and 1 digital manipulation) [224]. To the best of our knowledge, we are the first to attempt detecting 25 attack types across 3 categories.

Multi-task Learning. In multi-task learning (MTL), a task, $\mathcal{T}_i$ is usually accompanied by a training dataset, $\mathcal{D}_{tr}$ consisting of N_t training samples, *i.e.*, $D_{tr} = \{\mathbf{x}_i^{tr}, y_i^{tr}\}_{i=1}^{N_{tr}}$, where $\mathbf{x}_i^{tr} \in \mathbb{R}$ is the ith training sample in $\mathcal{T}_i$ and y_i^t is its label. Most MTL methods rely on well-defined tasks [226–228]. Crawshaw *et al.* [229] summarize various works on MLT with CNNs. In this work, we propose

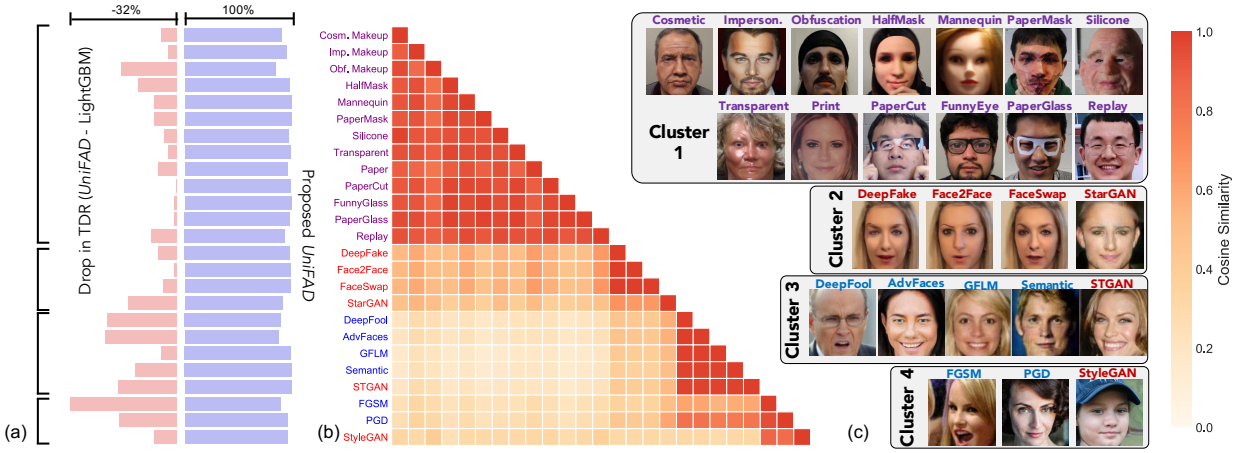


Figure 4.2 (a) Detection performance (TDR @ 0.2% FDR) in detecting each attack type by the proposed *UniFAD* (purple) and the difference in TDR from the best fusion scheme, LightGBM [36] (pink). (b) Cosine similarity between mean features for 25 attack types extracted by *JointCNN*. (c) Examples of attack types from 4 different clusters via k-means clustering on JointCNN features. Attack types in purple, blue, and red denote spoofs, adversarial, and digital manipulation attacks, respectively.

a MTL framework in an extreme situation where only a single task is available (bona fide vs. 25 attack types) and utilize k-means clustering to construct new auxiliary tasks from $\mathcal{D}_{tr}$. A recent study also utilized k-means for constructing new tasks, however, their approach utilizes a meta-learning framework where the groups themselves can alter throughout training [230]. We show that this is problematic for face attacks since attacks that share similar characteristics should be trained jointly. Instead, we propose a new unified attack detection framework that first utilizes k-means to partition the 25 attacks types, and then learns shared and attack-specific representations to distinguish them from bona fides.

4.3 Dissecting Prevailing Defense Systems

4.3.1 Datasets

In order to detect 25 attack types (6 adversarial, 6 digital manipulation, and 13 spoofs), we propose the *GrandFake* dataset, an amalgamation of multiple face attack datasets from each category. We

provide additional details of *GrandFake* in Sec. 4.5.1.

4.3.2 Drawback of JointCNN

Consider the diversity in the available attacks: from imperceptible adversarial perturbations to digital manipulation attacks, both of which are entirely different from physical print attacks (hard surface, glossy, 2D). Even within the spoof category, characteristics of mask attacks are quite different from replay attacks. In addition, discriminative cues for some attack types may be observed in high-frequency domain (*e.g.*, defocused blurriness, chromatic moment), while others exhibit low-frequency cues (*e.g.*, color diversity and specular reflection). For these reasons, learning a common feature space to discriminate all attack types from bona fides is challenging and a JointCNN may fail to generalize well even on attack types seen during training.

We demonstrate this by first training a JointCNN on the 25 attack types in *GrandFake* dataset. We then compute an *attack similarity matrix* between the 25 types (see Fig. 4.2(b)). The mean feature for each attack type is first computed on a validation set composed of $1,000$ images per attack. We then compute the pairwise cosine similarity between mean features from all attack pairs. From Fig. 4.2, we note that physical attacks have little correlation with adversarial attacks and therefore, learning them jointly within a common feature space may degrade detection performance.

Although prevailing JointCNN-based defense achieve near perfect detection when trained and evaluated on the respective attack types in isolation, we observe a significantly degraded performance when trained and tested on all 3 attack categories together (see Tab. 4.2). In other words, even when a prevailing SOTA defense system is trained on all 3 categories, it may lead to degraded performance on testing.

4.3.3 Unifying Multiple JointCNNs

Another possible approach is to consider ensemble techniques; instead of using a single JointCNN detector, we can fuse decisions from multiple individual detectors that are already experts in distinguishing between bona fides and attacks from their respective attack category. Given three SOTA

detectors, one per attack category, we perform a comprehensive evaluation on parallel and sequential score-level fusion schemes.

In our experiments, we find that, indeed, fusing score-level decisions from single-category detectors outperforms a single SOTA defense system trained on all attack types. Note that efforts in utilizing prevailing defense systems rely on the assumption that attack categories are independent of each other. However, Fig. 4.2 shows that some digital manipulation attacks, such as STGAN and StyleGAN, are *more closely related* to some of the adversarial attacks (*e.g.*, AdvFaces, GFLM, and Semantic) than other digital manipulation types. This is likely because all five methods utilize a GAN to synthesize their attacks and may share similar attack characteristics. Therefore, a SOTA adversarial detector and a SOTA digital manipulation detector may individually excel at their respective categories, but may not provide complementary decisions when fused. Instead of training detectors on groups with manually assigned semantics (*e.g.*, adversarial, digital manipulation, spoofs), it is better to train JointCNNs on coherent attacks. In addition, utilizing decisions from pre-trained JointCNNs may tend to overfit to the attack categories used for training.

4.4 Proposed Method

We propose a new multi-task learning framework for Unified Attack Detection, namely *UniFAD*, by training an end-to-end network for improved physical and digital face attack detection. In particular, a k-means augmentation module is utilized to automatically construct auxiliary tasks to enhance single task learning (such as a JointCNN). Then, a joint model is decomposed into a feature extractor (shared layers) $\mathcal{F}$ that is shared across all tasks, and task-specific branches for each auxiliary task. Fig. 4.3 illustrates the auxiliary task creation and the training process of *UniFAD*.

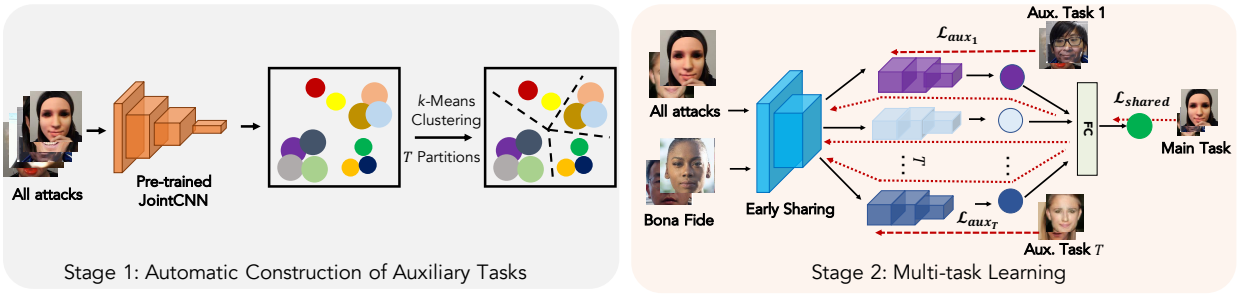


Figure 4.3 An overview of training *UniFAD* in two stages. Stage 1 automatically clusters coherent attack types into T groups. Stage 2 consists of a MTL framework where early layers learn generic attack features while T branches learn to distinguish bona fides from coherent attacks.

4.4.1 Problem Definition

Let the "main task" be defined as the overall objective of a unified attack detector: given an input image, $\mathbf{x}$, assign a score close to 0 if $\mathbf{x}$ is bona fide or close to 1 if $\mathbf{x}$ is any of the available face attack types. We are also given a labeled training set, D_{tr}. Prevailing defenses follow a single task learning approach where the main task is adopted to be the ultimate training objective. In order to avoid the shortcomings of a JointCNN and unification of multiple JointCNNs, we first use D_{tr} to automatically construct multiple auxiliary tasks $\{\mathcal{T}_t\}_{t=1}^{T}$, where T_i is the ith cluster of coherent attack types. If the auxiliary tasks are appropriately constructed, jointly learning these tasks along with the main task should improve unified attack detection compared to a single task learning approach.

4.4.2 Automatic Construction of Auxiliary Tasks

One way to construct auxiliary tasks is to train a separate binary JointCNN on each attack type. Such partitioning massively increases computational burden (*e.g.*, training and testing 25 JointCNNs). Other simple partitioning methods, such as randomly partition are likely to cluster uncorrelated attacks. On the other hand, clustering in the pixel-space is also unappealing due to poor correlation between the distances in the pixel-space, and clustering in the high-dimensional space is challenging [231]. Therefore, we require a reasonable alternative to manual inspection of the attack similarity matrix in Fig. 4.2 to partition the attack types into appropriate clusters.

Fortunately, we already have a JointCNN trained via a single task learning framework that can extract salient representations. Thus, we can map the data $\{\mathbf{x}\}$ into JointCNN's embedding space $\mathcal{Z}$, producing $\{\mathbf{z}\}$. We can then utilize a traditional clustering algorithm, k-means, which takes a set of feature vectors as input, and clusters them into k distinct groups based on a geometric constraint. Specifically, for each attack type, we first compute the mean feature. We then utilize k-means clustering to partition the L features into $T(\leq L)$ sets, $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_T\}$ such that within-cluster sum of squares (WCSS) is minimized,

$$\arg\min_{\mathcal{P}} \sum_{i=1}^{T} \sum_{\bar{\mathbf{z}} \in \mathcal{P}_i} ||\bar{\mathbf{z}} - \mu_i||^2, \tag{4.4.1}$$

where, $\bar{\mathbf{z}}$ represents a mean feature for an attack type and μ_i is the mean of the features in $\mathcal{P}_i$. Fig. 4.2(c) shows an example on clustering the 25 attack types of *GrandFake*.

4.4.3 Multi-Task Learning with Constructed Tasks

With a multi-task learning framework, we learn coherent attack types jointly, while uncorrelated attacks are learned in their own feature spaces. We construct T "branches" where each branch is a neural network trained on a binary classification problem (*i.e.*, aux. task). The learning objective of each branch, $\mathcal{B}_t$, is to minimize,

$$\mathcal{L}_{aux_t} = \mathbb{E}_{\mathbf{x}} \left[\log \mathcal{B}_t(\mathbf{x}_{bf})\right] + \mathbb{E}_{\mathbf{x}} \left[\log \left(1 - \mathcal{B}_t(\mathbf{x}_{fake}^{\mathcal{P}_t})\right)\right]. \tag{4.4.2}$$

where $\mathbf{x}_{bf}$ denotes bona fide images and $\mathbf{x}_{fake}^{\mathcal{P}_t}$ is face attacks corresponding to the attack types in the partition $\mathcal{P}_t$.

4.4.4 Parameter Sharing

Early Sharing. We adopt a hard parameter sharing module which learns a common feature representation for distinguishing between bona fides and attacks prior to aux. task learning branches. Baxter [232] demonstrated the shared parameters have a lower risk of overfitting than the task-specific parameters. Therefore, adopting early convolutional layers as a pre-processing step prior

to branching can help *UniFAD* in its generalization to all 3 categories. We construct hidden layers between the input and the branches to obtain shared features, $\mathbf{h} = \mathcal{F}(\mathbf{x})$, while the auxiliary learning branches output $\mathcal{B}_t(\mathbf{h})$.

Late Sharing. Each branch $\mathcal{B}_t$ is trained to output a decision score where scores closer to 0 indicate that the input image is a bona fide, whereas, scores closer to 1 correspond to attack types pertaining to the branch's partition. The scores from all T branches are then concatenated and passed to a final decision layer. For simplicity, we define the final decision output as, $FC(\mathbf{x}) := FC(\mathcal{B}_1(\mathbf{h}), \mathcal{B}_2(\mathbf{h}), \ldots, \mathcal{B}_T(\mathbf{h}))$.

The early shared layers and the final decision layer are learned via a binary cross-entropy loss,

$$\mathcal{L}_{shared} = \mathbb{E}_{\mathbf{x}}\left[\log FC(\mathbf{x}_{bf})\right] + \mathbb{E}_{\mathbf{x}}\left[\log\left(1 - FC(\mathbf{x}_{fake})\right)\right], \tag{4.4.3}$$

between bona fides and all available attack types.

4.4.5 Training and Testing

The entire network is trained in an end-to-end manner by minimizing the following composite loss,

$$\mathcal{L}_{UniFAD} = \mathcal{L}_{shared} + \sum_{t=1}^{T} \mathcal{L}_{aux_t}. \tag{4.4.4}$$

The $\mathcal{L}_{shared}$ loss is backpropagated throughout *UniFAD*, while $\mathcal{L}_{aux_t}$ is only responsible for updating the weights of the branch, $\mathcal{B}_t$, and the final classification layer. For the forward and backward passes of $\mathcal{L}_{shared}$, an equal number of bona fide and attack samples are used for training. On the other hand, for training each branch, $\mathcal{B}_t$, we sample the equal number of bona fides and equal number of attack images from the attack partition $\mathcal{P}_t$.

Attack Detection. During testing, an image passes through the shared layers and then each branch of *UniFAD* outputs a decision whether the image is bona fide (values close to 0) or an attack (close to 1). The final decision layer outputs the final decision score. Unless stated otherwise, we use the final decision scores to report performance.

Attack Classification. Once an attack is detected, *UniFAD* can automatically classify the attack

type and category. For all L attack types in the training set, we extract intermediate 128-dim feature vectors from T branches. The features are then concatenated and the mean feature across all L attack types is computed, such that, we have L feature vectors of size $T \times 128$. For a detected attack, Cosine similarity is computed between the testing sample's feature vector and the mean training features for L types. The predicted attack type is the one with the highest similarity score.

4.5 Experimental Results

4.5.1 Experimental Settings

Dataset. *GrandFake* consists of 25 face attacks from 3 attack categories. Both bona fide and fake faces are of varying quality due to different capture conditions.

Bona Fide Faces. We utilize faces from CASIA-WebFace [10], LFW [8], CelebA [17], SiW-M [1], and FFHQ [18] datasets since the faces therein cover a broad variation in race, age, gender, pose, illumination, expression, resolution, and acquisition conditions.

Adversarial Faces. We craft adversarial faces from CASIA-WebFace [10] and LFW [8] via 6 SOTA adversarial attacks: FGSM [34], PGD [219], DeepFool [141], AdvFaces [16], GFLM [32], and SemanticAdv [186]. These attacks were chosen for their success in evading SOTA AFR systems such as ArcFace [9].

Digital Manipulation. There are four broad types of digital face manipulation: identity swap, expression swap, attribute manipulation, and entirely synthesized faces [20]. We use all clips from FaceForensics++ [77], including identity swap by FaceSwap and DeepFake[1], and expression swap by Face2Face [76]. We utilize two SOTA models, StarGAN [78] and STGAN [79], to generate attribute manipulated faces in Celeba [17] and FFHQ [18]. We use the pretrained StyleGAN2 model[2] to synthesize 100K fake faces.

Physical Spoofs. We utilize the publicly available *SiW-M* dataset [1], comprising 13 spoof

[1]https://github.com/deepfakes/faceswap
[2]https://github.com/NVlabs/stylegan2

	TDR (%) @ 0.2% FDR	Year	Proposed For	Adv.	Dig. Man.	Phys.	Overall	Time (ms)
w/o Re-train	FaceGuard [19]	2020	Adversarial	99.91	22.28	00.58	29.64	01.41
	FFD [20]	2020	Digital Manipulation	09.49	94.57	01.25	34.55	11.57
	SSRFCN [21]	2020	Spoofs	00.25	00.76	93.19	22.71	02.22
	MixNet [233]	2020	Spoofs	00.36	09.83	78.21	21.12	12.47
Baselines	FaceGuard [19]	2020	Adversarial	99.86	41.56	04.35	56.69	01.41
	FFD [20]	2020	Digital Manipulation	76.06	91.32	87.43	68.25	11.57
	SSRFCN [21]	2020	Spoofs	08.23	27.67	89.19	43.26	02.22
	One-class [61]	2020	Spoofs	04.81	45.96	79.32	39.40	07.92
	MixNet-*UniFAD*	2021	All	82.33	91.59	94.60	90.07	12.47
Fusion Schemes	Cascade [40]	–	–	88.39	81.98	69.19	77.46	05.16
	Min-score	–	–	03.65	11.08	00.43	07.22	16.14
	Median-score	–	–	10.87	42.33	47.19	39.48	16.12
	Mean-score	–	–	14.53	47.18	61.32	38.23	16.12
	Max-score	–	–	85.32	61.93	56.87	73.89	16.13
	Sum-score	–	–	74.93	58.01	50.34	69.21	16.11
	LightGBM [36]	–	–	76.25	81.28	88.52	85.97	17.92
	Proposed UniFAD	2021	All	92.56	97.21	98.76	94.73	02.59

Table 4.2 Detection accuracy (TDR (%) @ 0.2% FDR) on *GrandFake* dataset. Results on fusing FaceGuard [19], FFD [20], and SSRFCN [21] are also reported. We report time taken to detect a single image (on a Nvidia 2080Ti GPU).

types. Compared with other spoof datasets (Tab. 4.1), SiW-M is diverse in spoof attack types, environmental conditions, and face poses.

Protocol. As is common practice in face recognition literature, bona fides and attacks from CASIA-WebFace [10] are used for training, while bona fides and attacks for LFW [8] are sequestered for testing. The bona fides and attacks from other datasets are split into 70% training, 5% validation, and 25% testing.

Implementation. *UniFAD* is implemented in Tensorflow, and trained with a constant learning rate of $(1e^{-3})$ with a mini-batch size of 180. The objective function, $\mathcal{L}_{UniFAD}$, is minimized using Adam optimizer for $100K$ iterations. Data augmentation during training involves random horizontal flips with a probability of 0.5.

Metrics. Studies on different attack categories provide their own metrics. Following the recommendation from IARPA ODIN program, we report the TDR @ 0.2% FDR[3] and the overall detection accuracy (in Addendum (Sec. 4.6).

[3]https://www.iarpa.gov/index.php/research-programs/odin

4.5.2 Comparison with Individual SOTA Detectors

In this section, we compare the proposed *UniFAD* to prevailing face attack detectors via publicly available repositories provided by the authors (see Addendum (Sec. 4.6).

Without Re-training. In Tab. 4.2, we first report the performance of 4 pre-trained SOTA detectors. These baselines were chosen since they report the best detection performance in datasets with largest numbers of attack types in their respective categories (see Tab. 4.1). We also report the performance of a generalizable spoof detector with sub-networks, namely *MixNet*. MixNet semantically assigns spoofs into groups (print, replay, and masks) for training each sub-network without any shared representation. We find that pre-trained methods indeed excel in their specific attack categories, however, generalization performance across all 3 categories deteriorates catastrophically.

With Re-training. After re-training the 4 SOTA detectors on all 25 attack types, we find that they generalize better across categories. FaceGuard [19], FFD [20], SSRFCN [21], and One-Class [61], all employ a JointCNN for detecting attacks. Unsurprisingly, these defenses perform well on some attack categories, while failing on others.

For a fair comparison, we also modify MixNet, namely *MixNet-UniFAD* such that clusters are assigned via k-means with 4 branches. In contrast to *MixNet-UniFAD*, *UniFAD* (i) employs early shared layers for generic attack cues, and (ii) each branch learns to distinguish between bona fides and specific attack types. MixNet, on the other hand, assigns a bona fide label (0) to attack types outside a respective branch's partition. This negatively impacts network convergence. Overall, we find that *UniFAD* outperforms *MixNet-UniFAD* with TDR $90.07\% \rightarrow 94.73\%$ @ 0.2% FDR.

4.5.3 Comparison with Fused SOTA Detectors

We also comprehensively evaluate detection performance on fusing SOTA detectors. We utilize three best performing detectors from each attack category, namely FaceGuard [19], FFD [20], and SSRFCN [21]. Inspired by the Viola-Jones object detection [40], we adopt a sequential ensemble

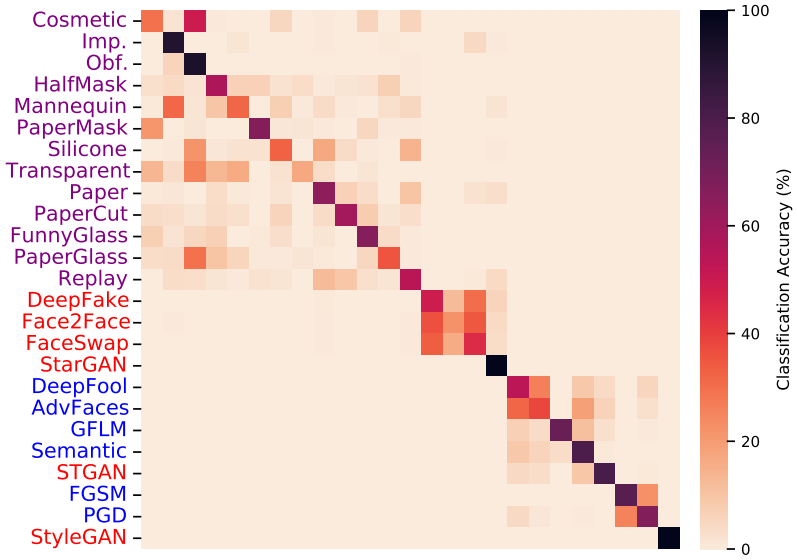


Figure 4.4 Confusion matrix representing the classification accuracy of *UniFAD* in identifying the 25 attack types. Majority of misclassifications occur within the attack category. Darker values indicate higher accuracy. Overall, *UniFAD* achieves 75.81% and 97.37% classification accuracy in identifying attack types and categories, respectively. Purple, blue, and red denote spoofs, adversarial, and digital manipulation attacks, respectively.

technique, namely Cascade [40], where an input probe is passed through each detector sequentially. We also evaluate 5 parallel score fusion rules (min, mean, median, max, and sum) and a SOTA ensemble technique, namely LightGBM [36]. More details are provided in Addendum (Sec. 4.6. Indeed, we observe an overhead in detection speed compared to the individual detectors in isolation, however, cascade, max-score fusion and LightGBM [36] can enhance the overall detection performance compared to the individual detectors at the cost of slower inference speed. Since the individual detectors still train with incoherent attack types, we find that proposed *UniFAD* outperforms all the considered fusion schemes.

In Fig. 4.2(a), we show the performance degradation of LightGBM [36], the best fusing baseline, w.r.t. *UniFAD*. We observe that among 4 clusters, the last 2 have the overall largest degradation. Interestingly, these 2 clusters are the only ones including attack types across different attack categories, learned via our k-mean clustering. In other words, the cross-category attacks types within a branch benefit each other, leading to the largest performance gain over [36]. This further demonstrates the necessity and importance of a unified detection scheme — the more attack types

the detector sees, the more likely it would nourish among each other and be able to generalize.

4.5.4 Attack Classification

We classify the exact attack type and categories using the method described in Sec. 4.4.5. In Fig. 4.4, we find that *UniFAD* can predict the attack type with 75.81% classification accuracy. While predicting the exact type may be challenging, we highlight that majority of the misclassifications occurs within attack's category. Without human intervention, once *UniFAD* is deployed in AFR pipelines, it can predict whether an input image is adversarial, digitally manipulated, or contains spoof artifacts with 97.37% accuracy.

4.5.5 Analysis of UniFAD

Architecture. We first ablate and analyze our architecture.

Ratio of Shared Layers. Our backbone network consists of a 4-layer CNN. In Fig. 4.5a, we report the detection performance when we incorporate 0, 1 (25%), 2 (50%), 3 (75%), and 4 (100%) layers for early sharing. We observe a trade-off between detection performance and the number of early layers: too many reduces the effects of learning task-specific features via branching, whereas, less number of shared layers inhibits the network from learning generic features that distinguish any attack from bona fides. We find that an even split results in superior detection performance.

Number of Branches. In Fig. 4.5b, we vary the number of branches (aux. tasks constructed via kMeans) and report the detection performance. Indeed, increasing the number of branches via additional clusters enhances detection performance. However, the performance saturates after 4 branches. *UniFAD* with 4 branches achieves TDR = 94.73% @ 0.2% FDR, whereas, 5 and 6 branches achieve TDRs of 94.33 and 94.62 at 0.2%, respectively. For $T = 5$, we notice that StyleGAN is isolated from Cluster 3 (see Fig. 4.2(c)) into a separate cluster. Learning to discriminate StyleGAN separately may offer no significant advantage than learning jointly with FGSM and PGD. Thus, we choose $T = 4$ due to lower network complexity and higher inference efficiency.

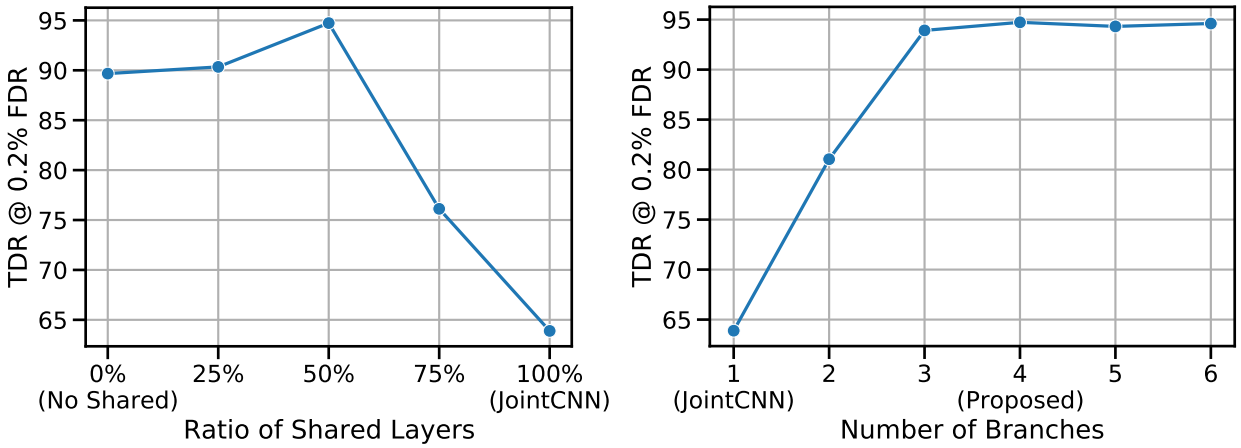


Figure 4.5 Detection performance with respect to varying ratio of shared layers (left) and number of branches (right). Our proposed architecture uses 50% shared layers with 4 branches.

Model	Modules			Overall
	Shared Layers	Branching	kMeans	TDR (%) @ 0.2% FDR
JointCNN	✓			63.89
$\mathcal{B}_{Semantic}$		✓		86.17
$\mathcal{B}_{Random}$		✓		53.95 ± 08.02
$\mathcal{B}_{kMeans}$		✓	✓	89.67
SharedSemantic	✓	✓		92.44
Proposed	✓	✓	✓	**94.73**

Table 4.3 Ablation study over components of *UniFAD*. Branching via "$\mathcal{B}_{Semantic}$", "$\mathcal{B}_{Random}$", and "$\mathcal{B}_{kMeans}$" refer to partitioning attack types by their semantic categories, randomly, and kMeans. "SharedSemantic" includes shared layers prior to branching.

Branch Generalizability. In Fig. 4.6, scores from the 4 branches are used to compute the detection performance on attack types within respective partitions and those outside a branch's partition (see Fig. 4.2(b)). Since attack types outside a branch's partition are purportedly incoherent, we see a drop in performance; validating the drawback of JointCNN. We find that the lowest performance branch, Branch 4, also exhibits the best generalization performance across other attack types. This is likely because learning to distinguish bona fides from imperceptible perturbations from FGSM, PGD, and minute synthetic noises from StyleGAN yields a tighter decision boundary which may contribute to better generalization across digital attacks. Anti-spoofing (Branch 1) itself does not directly aid in detecting digital attacks.

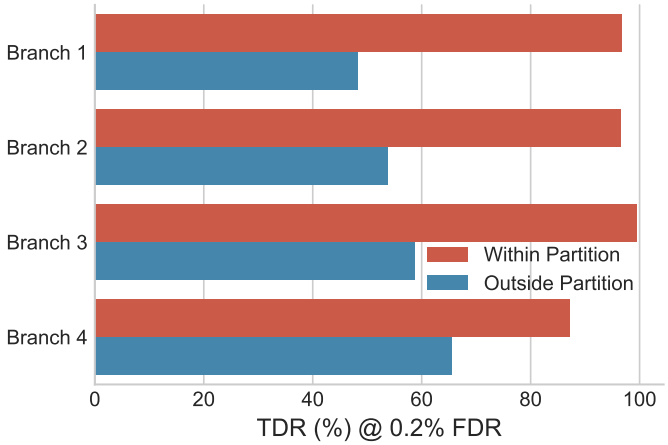


Figure 4.6 Detection performance on attack types within and outside a branch's partition. Performance drops on attacks outside partition as they may not have any correlation with within-partition attack types.

Ablation Study. In Tab. 4.3, we conduct a component-wise ablation study over *UniFAD*. We study different partitioning techniques to group the 25 attack types. We employ semantic partitioning, $\mathcal{B}_{Semantic}$ where attack types are clustered into the 3 categories. Another technique is to split the 25 attack types into 4 clusters randomly, $\mathcal{B}_{Random}$. We report the mean and standard deviation across 3 trials of random splitting. We also report the performance of clustering via kMeans. We find that both $\mathcal{B}_{Semantic}$ and $\mathcal{B}_{kMeans}$ outperforms JointCNN. Thus, learning separate feature spaces via MTL for disjoint attack types can improve overall detection compared to a JointCNN. We als find that incorporating early shared layers into $\mathcal{B}_{Semantic}$, namely $\mathcal{B}_{SharedSemantic}$, can further improve detection from $86.17\% \rightarrow 92.44\%$ TDR @ 0.2% FDR. However, as we observed in Fig. 4.2, even within semantic categories, some attack types may be incoherent. By automatic construction of auxiliary tasks with k-means clustering and shared representation (Proposed), we can further enhance the detection performance to TDR = 94.73% @ 0.2% FDR.

Failure Cases. Fig. 4.7 shows a few failure cases. Majority of the failure cases for digital attacks are due to imperceptible perturbations. In contrast, failure to detect spoofs can likely be attributed to the subtle nature of transparent masks, blurring, and illumination changes.

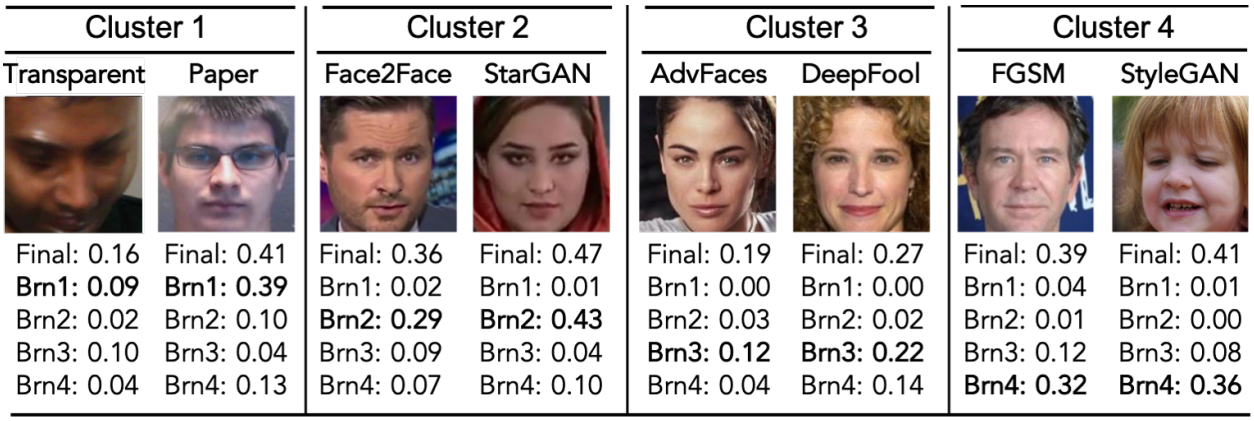


Figure 4.7 Example cases where *UniFAD* fails to detect face attacks. Final detection scores along with scores from each of the four branches ($\in [0, 1]$) are given below each image. Scores closer 0 indicate bona fide. Branches responsible for the respective cluster are highlighted in bold.

4.6 Addendum

Here, we provide additional details on the proposed *GrandFake* dataset and *UniFAD* and baselines.

4.6.1 GrandFake Dataset

The *GrandFake* dataset is composed of several widely adopted face datasets for face recognition, face attribute manipulation, and synthesis. Specific details on the source datasets along with training and testing splits are provided in Tab. 4.4. We ensure that there is no identity overlap between any of the training and testing splits as follows:

1. We removed 84 subjects in CASIA-WebFace that overlap with LFW. In addition, CASIA-WebFace is only used for training, while LFW is only used for testing.

2. SiW-M comprises of high-resolution photos of non-celebrity subjects without any identity overlap with other datasets. Training and testing splits are composed of videos pertaining to different identities.

3. Identities in CelebA training set are different than those in testing.

4. FFHQ comprises of high-resolution photos of non-celebrity subjects on Flicker. FFHQ is utilized solely for bona fides in order to add diversity to the quality of face images.

4.6.2 Implementation Details

All the models in the paper are implemented using Tensorflow r1.12. A single NVIDIA GeForce GTX 2080Ti GPU is used for training *UniFAD* on *GrandFake* dataset.

Preprocessing All face images are first passed through MTCNN face detector [41] to detect 5 facial landmarks (two eyes, nose and two mouth corners). Then, similarity transformation is used to normalize the face images based on the five landmarks. After transformation, the images are resized to 160×160. Before passing into *UniFAD* and baselines, each pixel in the RGB image is normalized $\in [-1, 1]$ by subtracting 128 and dividing by 128. **All the testing images in this chapter are from the identities in the test dataset.**

Network Architecture The backbone network, JointCNN, comprises of a 4-layer binary CNN:

- `d32,d64,d128,d256,fc128,fc1`,

where `dk` denotes a 4×4 convolutional layer with k filters and stride 2 and `fcN` refers to a fully-connected layer with N neuron outputs.

The proposed *UniFAD* with 4 branches is composed of:

- Early Layers:

  `d32,d64`,

- Branch 1:

  `d128,d256,fc128,fc1`,

- Branch 2:

  `d128,d256,fc128,fc1`,

- Branch 3:

  `d128,d256,fc128,fc1`,

- Branch 4:

  `d128,d256,fc128,fc1`,

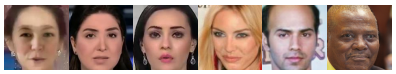
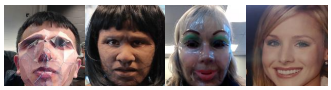
			# Total Samples	# Training	# Validation	# Testing	Examples
Datasets	Bona Fides	CASIA [10]	10,575	10,075	500	0	
		CelebA [17]	193,506	134,954	500	58,052	
		LFW [8]	9,164	0	0	9,164	
		FFHQ [18]	20,999	14,200	500	6,299	
		SiW-M [1]	107,494	74,746	500	32,248	CASIA [10] CelebA [17] LFW [8] FFHQ [18] SiW-M [1]
		Total	**341,738**	**233,975**	**2,000**	**105,763**	
Attacks	Adversarial	FGSM	19,739	10,495	80	9,164	
		PGD	19,739	10,495	80	9,164	
		DeepFool	19,739	10,495	80	9,164	
		AdvFaces	19,739	10,495	80	9,164	
		GFLM	17,946	8,702	80	9,164	FGSM PGD DeepFool AdvFaces GFLM Semantic
		Semantic	19,739	10,495	80	9,164	
	Digital Manipulation	DeepFake	18,165	10,393	80	7,692	
		Face2Face	18,204	10,385	80	7,739	
		FaceSwap	14,492	8,299	80	6,113	
		STGAN	29,983	9,903	80	20,000	
		StarGAN	45,473	10,406	80	34,987	DeepFake F2F FaceSwap StarGAN STGAN StyleGAN
		StyleGAN	76,604	10,919	80	65,605	
	Spoofs	Cosmetic	2,638	1,766	80	792	
		Impersonation	9,184	6,348	80	2,756	
		Obfuscation	3,611	2,447	80	1,084	
		HalfMask	10,486	7,260	80	3,146	Cosm. Imp. Obf. HalfMask Mann.
		Mannequin	5,287	3,620	80	1,587	
		PaperMask	2,550	1,705	80	765	
		Silicone	5,038	3,446	80	1,512	
		Transparent	11,451	7,935	80	3,436	PaperMask Silicone Trans. Print
		Print	10,530	7,290	80	3,160	
		PaperCut	13,178	9,144	80	3,954	
		FunnyEye	23,470	16,349	80	7,041	
		PaperGlass	18,563	12,914	80	5,569	PaperCut Funnyeye PaperGlass Replay
		Replay	12,126	8,408	80	3,638	
		Total	**447,674**	**210,114**	**2,000**	**235,560**	

Table 4.4 Composition and statistics for the proposed *GrandFake* dataset. We also include the evaluation protocol for the seen attack scenario.

4.6.3 Digital Attack Implementation

Adversarial attacks are synthesized via publicly available author codes:

- FGSM/PGD/DeepFool: https://github.com/tensorflow/cleverhans
- AdvFaces: https://github.com/ronny3050/AdvFaces

- GFLM: https://github.com/alldbi/FLM
- SemanticAdv: https://github.com/AI-secure/SemanticAdv

Digital manipulation attacks are also generated via publicly available author codes:

- DeepFake/Face2Face/FaceSwap: https://github.com/ondyari/FaceForensics/tree/original
- STGAN: https://github.com/csmliu/STGAN
- StarGAN: https://github.com/yunjey/stargan
- StyleGAN-v2: https://github.com/NVlabs/stylegan2

4.6.4 Baseline Implementation

Individual Detectors. We evaluate all *individual* defense methods, except MixNet [233], via publicly available repositories provided by the authors. We provide the public links to the author codes below:

- FaceGuard [19]: https://github.com/ronny3050/FaceGuard
- FFD [20]: https://github.com/JStehouwer/FFD_CVPR2020
- SSRFCN [21]: https://github.com/ronny3050/SSRFCN
- One-Class [61]: https://github.com/anjith2006/bob.paper.oneclass_mccnn_2019

Fusion of JointCNNs. In our work, we employ 5 parallel score-level fusion rules. For a testing input image, we extract three scores (in $[0, 1]$) from three SOTA individual detectors (FaceGuard [19], FFD [20], and SSRFCN [21]). The final decision score is computed via the fusion rule operator, namely *min*, *mean*, *median*, *max*, and *sum*. LightGBM [36] is a tree-ensemble learning method where a Gradient Boosted Decision Tree is trained from the three scores (from individual SOTA dectors) to output to the final decision. We use Microsoft's LightGBM implementation: https://github.com/microsoft/LightGBM. **We train the LightGBM model on the training set of *GrandFake*.**

	Method	Year	Proposed For	Metric	Adv.	Dig. Man.	Phys.	Overall
w/o Re-train	FaceGuard [19]	2020	Adversarial	TDR	99.91	22.28	00.58	29.64
				Acc.	99.78	67.12	51.02	71.03
	FFD [20]	2020	Digital Manipulation	TDR	09.49	94.57	01.25	34.55
				Acc.	56.42	97.87	53.42	75.29
	SSRFCN [21]	2020	Spoofs	TDR	00.25	00.76	93.19	22.71
				Acc.	50.01	50.93	96.12	69.11
	MixNet [233]	2020	Spoofs	TDR	00.36	09.83	78.21	21.12
				Acc.	50.43	55.98	85.47	61.26
Baselines	FaceGuard [19]	2020	Adversarial	TDR	99.86	41.56	04.35	56.69
				Acc.	99.71	71.23	54.06	81.88
	FFD [20]	2020	Digital Manipulation	TDR	76.06	91.32	87.43	68.25
				Acc.	87.15	93.40	91.37	89.06
	SSRFCN [21]	2020	Spoofs	TDR	08.23	27.67	89.19	43.26
				Acc.	54.02	69.18	90.91	83.41
	One-class [61]	2020	Spoofs	TDR	04.81	45.96	79.32	39.40
				Acc.	53.99	64.08	86.65	80.74
	MixNet-*UniFAD*	2021	All	TDR	82.33	91.59	94.60	90.07
				Acc.	89.32	94.50	96.18	93.19
Fusion Schemes	Cascade [40]	-	–	TDR	88.39	81.98	69.19	77.46
				Acc.	91.33	89.17	79.92	85.16
	Min-score	-	–	TDR	03.65	11.08	00.43	07.22
				Acc.	51.61	66.76	50.88	55.62
	Median-score	-	–	TDR	10.87	42.33	47.19	39.48
				Acc.	55.12	59.58	57.44	59.22
	Mean-score	-	–	TDR	14.53	47.18	61.32	38.23
				Acc.	55.69	54.19	73.87	55.92
	Max-score	-	–	TDR	85.32	61.93	56.87	73.89
				Acc.	89.26	68.11	60.08	69.43
	Sum-score	-	–	TDR	74.93	58.01	50.34	69.21
				Acc.	83.85	67.48	64.72	73.10
	LightGBM [36]	-	–	TDR	76.25	81.28	88.52	85.97
				Acc.	84.19	89.46	94.56	90.56
	Proposed UniFAD	2021	All	TDR	92.56	97.21	98.76	94.73
				Acc	95.18	98.32	98.96	96.89

Table 4.5 Detection performance (TDR (%) @ 0.2% FDR and Accuracy (%)) on *GrandFake* dataset under the *seen* attack scenario.

4.6.5 Seen Attacks

Tab. 4.5 reports the detection performance (TDR (%) @ 0.2% FDR and accuray (%)) of *UniFAD* and baselines on *GrandFake* dataset under the seen attack scenario. Training and testing splits are provided in Tab. 4.4. Overall, *UniFAD* outperforms all fusion schemes and baselines.

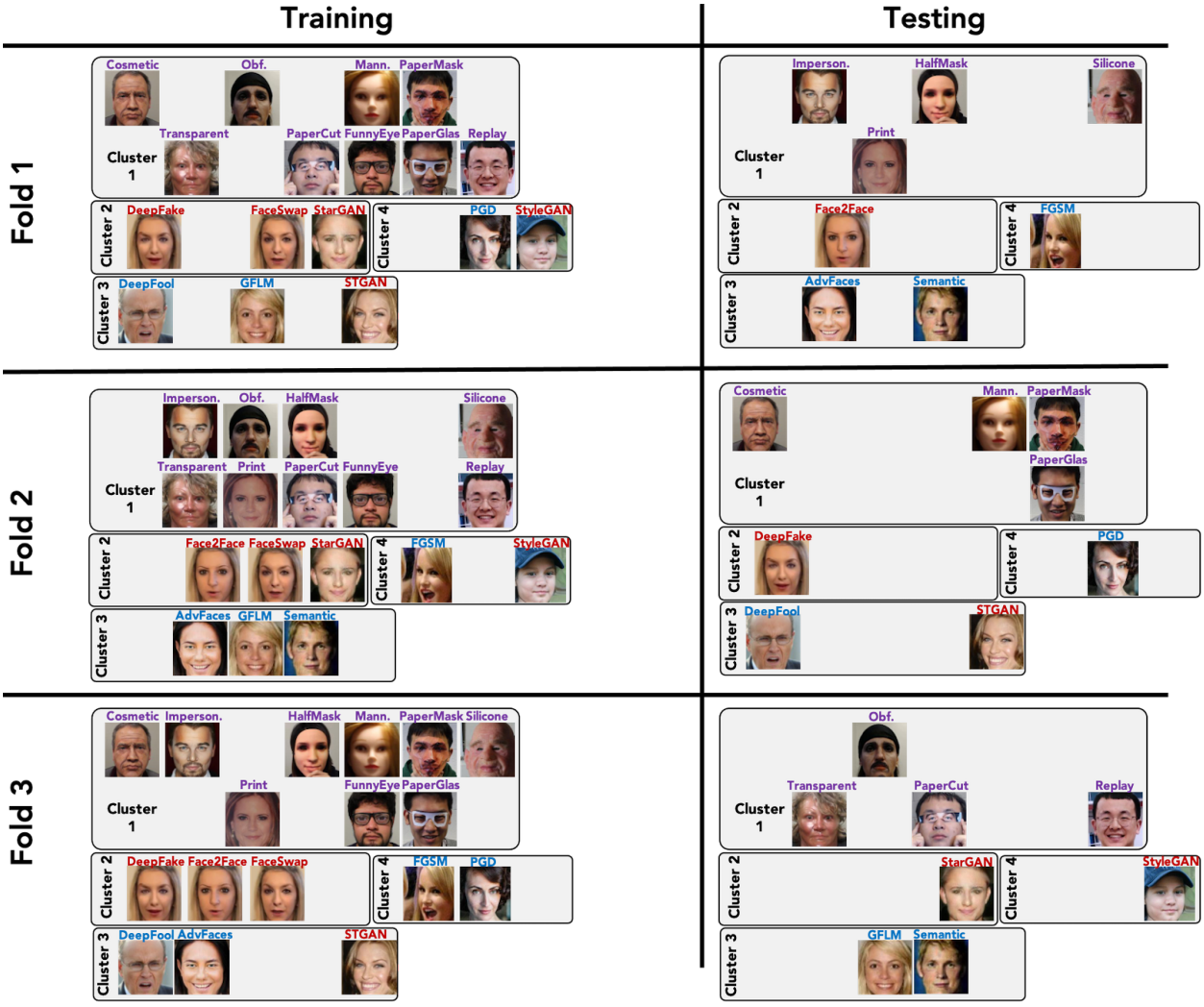


Figure 4.8 Training and testing splits for generalizability study.

Method	Metric (%)	Fold 1	Fold 2	Fold 3	Mean $\pm$ Std.
FaceGuard [19]	TDR	41.38	54.19	36.82	44.13 ± 9.01
	Acc.	58.42	64.19	55.74	59.45 ± 4.32
FFD [20]	TDR	53.19	62.45	52.94	56.20 ± 5.42
	Acc.	66.15	69.33	67.86	67.78 ± 1.59
SSRFCN [21]	TDR	49.10	64.92	61.18	58.84 ± 8.26
	Acc.	60.07	72.77	69.83	66.57 ± 6.64
MixNet- *UniFAD*	TDR	67.19	73.18	72.74	71.04 ± 3.33
	Acc.	75.64	79.40	78.73	77.93 ± 2.00
LightGBM	TDR	51.65	65.73	67.91	61.76 ± 8.83
	Acc.	69.34	73.66	75.80	72.93 ± 3.29
Proposed *UniFAD*	TDR	76.18	83.19	82.67	80.68 ± 3.91
	Acc.	85.35	89.62	85.88	86.95 ± 2.32

Table 4.6 Generalization performance (TDR (%) @ 0.2% FDR and Accuracy (%)) on *GrandFake* dataset under unseen attack setting. Each fold comprises of 8 unseen attacks from all 4 branches.

4.6.6 Generalizability to Unseen Attacks

Under this setting, we evaluate the generalization performance on 3 folds (see Fig. 4.8). The folds are computed as follows: we hold out $1/3$ of the total attack types in a branch for testing and the remaining are used for training. For *e.g.*, branch 1 consisting of 13 attack types are *randomly* split such that we test on 4 unseen attack types, while the remaining 9 attack types are used for training. We perform 3 folds of such random splitting. In total, each fold consists of 17 seen and 8 unseen attacks. For LightGBM, we utilize scores from FaceGuard [19], FFD [20], and SSRFCN [21] which are all trained only on the known attack types. We report the detection performance and the average and standard deviation across all folds in Tab. 4.6.

We find that branching-based methods, such as MixNet- *UniFAD* and the proposed *UniFAD*, significantly outperforms JointCNN-based methods such as FaceGuard [19], FFD [20], SSR-FCN [21], and LightGBM (fusion of the three). The superiority of the proposed *UniFAD* under

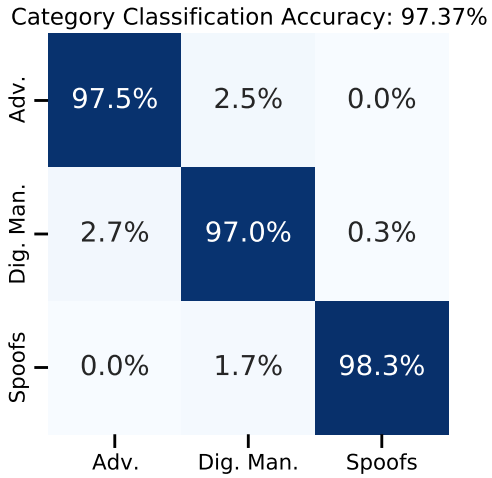


Figure 4.9 Confusion matrix representing the classification accuracy of *UniFAD* in identifying the 3 attack categories, namely adversarial faces, digital face manipulation, and spoofs. Majority of confusion occurs within digital attacks (adversarial and digital manipulation attacks).

unseen attacks is evident. By incorporating branches with coherent attacks, removing some attack types within a branch does not drastically affect the generalization performance.

In addition to superior generalization performance to unseen attack types, the proposed *UniFAD* also reduces the gap between seen and unseen attacks. Overall, the proposed *UniFAD* achieves 94.73% and 80.68% TDRs at 0.2% FDR under seen and unseen attack scenario. That is, we observe a relative reduction in TDR of 15% under unseen attacks, compared to the second best method, namely MixNet-*UniFAD*, which has a relative reduction in TDR of 22% under unseen attacks.

4.6.7 Attack Category Classification

In Fig. 4.9, we find that *UniFAD* can predict the attack category with 97.37% classification accuracy. We emphasize that majority of the misclassifications occurs within the digital attack space. That is, misclassifying adversarial attacks as digital manipulation attacks and vice-versa.

Among spoofs, 1.7% of them are misclassified as digital manipulation attacks. Majority of these are makeup attacks which have some correlation with some digital manipulation attacks such as DeepFake, Face2Face, and FaceSwap (see Fig. 2 in *main paper*). We posit that this likely because cosmetic and impersonation attacks apply makeup to eyebrows and cheeks which may

appear similar to ID-swapping methods such as DeepFake and Face2Face which also majorly alter the eyebrows and cheeks.

4.7 Summary

With new and sophisticated attacks being crafted against AFR systems in both digital and physical spaces, detectors need to be robust across all 3 categories. Prevailing methods indeed excel at detecting attacks in their respective categories, however, they fall short in generalizing across categories. While, ensemble techniques can enhance the overall performance, they still fail to meet the desired accuracy levels. Poor generalization can be predominantly attributed towards learning incoherent attacks jointly. With a new multi-task learning framework along with k-means augmentation, the proposed *UniFAD* achieved SOTA detection performance (TDR = 94.73% @ 0.2% FDR) on 25 face attacks across 3 categories. *UniFAD* can further identify categories with a 97.37% accuracy.

Chapter 5

Summary

This dissertation has addressed some important challenges that plague prevailing automated face recognition systems today. Our primary contribution lies in enhancing the robustness and security of any commodity face recognition system against malicious attacks such as presentation attacks (physically crafted spoofs), adversarial faces (imperceptible noises added to the input probe), and digital face manipulation attacks (identity and expression swapping, attribute manipulation, and entire face synthesis). All proposed solutions achieve state-of-the-art detection performance while maintaining high computational efficiency (< 4ms on Nvidia GTX 2080Ti GPU). We also make an effort to impart generalizability to unknown attack types that may be launched against an AFR pipeline in the future. In addition, all proposed detectors are interpretable such that AFR systems can be operated safely in covert scenarios. When a face attack is detected, authorities can be alerted with more than just a holistic "attack score". For instance, our proposed methods can highlight regions of the face image that contribute to the overall decision made by the detector. Lastly, since the exact attack types may not be known beforehand, our work is among the first to attempt detection across three attack categories in both physical and digital domains. We also presented a method to automatically classify the exact attack category whenever an attack is detected.

5.1 Contributions

Chapter 2 focused on safeguarding AFR systems from physical face spoofs. The contributions include:

- We show that features learned from local face regions have better generalization performance on detecting presentation attacks than those learned from the entire face image.

- We provide extensive experiments to show that the proposed presentation attack detection approach, outperforms other local region extraction strategies and state-of-the-art face presentation attack detection methods on one of the largest publicly available dataset, namely, SiW-M, comprised of 13 different presentation attack instruments. The proposed method reduces the Equal Error Rate (EER) by (i) 14% relative to state-of-the-art [88] under the unknown attack setting, and (ii) 40% on known presentation attack instruments. In addition, *SSR-FCN* achieves competitive performance on standard benchmarks on Oulu-NPU [2] dataset and outperforms prevailing methods on cross-dataset generalization (CASIA-FASD [3] and Replay-Attack [4]).

- The proposed presentation attack detection method is also shown to be more interpretable since it can directly predict the parts of the faces that are considered as presentation attacks.

In Chapter 3, we first designed an automated adversarial synthesis method. The contributions of the synthesis section are as follows:

- A GAN-based *AdvFaces* that learns to generate visually realistic adversarial face images that are misclassified by state-of-the-art AFR systems. Adversarial faces generated via AdvFaces are model-agnostic and transferable, and achieve high success rate on 5 state-of-the-art automated face recognition systems.

- Perceptual studies where human observers suggest that the adversarial examples appear more similar to the probe compared to the previous methods.

- Visualizing the facial regions, where pixels are perturbed and analyzing the transferability

of AdvFaces.

- An open-source automated adversarial face generator permitting users to control the amount of perturbation.

The latter part of the chapter focused on utilizing ideas present in the aforementioned adversarial synthesis method to detect any adversarial attack type. This work makes the following contributions:

- A new self-supervised framework, namely *FaceGuard*, for defending against adversarial face images. *FaceGuard* combines benefits of adversarial training, detection, and purification into a unified defense mechanism trained in an end-to-end manner.

- With the proposed diversity loss, a generator is regularized to produce stochastic and challenging adversarial faces. We show that the diversity in output perturbations is sufficient for improving *FaceGuard*'s robustness to unseen attacks compared to utilizing pre-computed training samples from known attacks.

- Synthesized adversarial faces aid the detector to learn a tight decision boundary around real faces. *FaceGuard*'s detector achieves SOTA detection accuracies of 99.81%, 98.73%, and 99.35% on 6 unseen attacks on LFW [8], Celeb-A [17], and FFHQ [18].

- As the generator trains, a purifier concurrently removes perturbations from the synthesized adversarial faces. With the proposed bonafide loss, the detector also guides purifier's training to ensure purified images are devoid of adversarial perturbations. At 0.1% False Accept Rate, *FaceGuard*'s purifier enhances the True Accept Rate of ArcFace [9] from 34.27% under no defense to 77.46%.

Lastly, the contributions of our study on unified detection of both physical and digital attacks in Chapter 4 are as follows:

- Among the first to define the task of face attack detection on 25 attack types across 3 attack categories: adversarial faces, digital face manipulation, and spoofs. We comprehensively

analyze the shortcomings of prevailing detectors. We also show that sequential and parallel ensemble learning can enhance detection compared to using a single SOTA detector.

- A novel **uni**fied **f**ace **a**ttack **d**etection framework, namely *UniFAD*, that automatically clusters similar attacks and employs a multi-task learning framework to detect digital and physical attacks.

- Proposed *UniFAD* achieves SOTA detection performance, TDR = 94.73% @ 0.2% FDR on a large fake face dataset, namely *GrandFake*. To the best of our knowledge, *GrandFake* is the largest face attack dataset studied in literature in terms of the number of diverse attack types.

- Proposed *UniFAD* allows for further identification of the attack categories, *i.e.*, whether attacks are adversarial, digitally manipulated, or contains physical spoofing artifacts, with a classification accuracy of 97.37%.

5.2 Suggestions for Future Work

The algorithms and models designed in this thesis are not limited to AFR systems. Attacks crafted in both physical and digital domains can also be launched against other biometric systems (such as automated fingerprint and iris recognition systems). The research presented in this dissertation can be extended in the following directions:

- **Adversarial Attacks on Other Biometrics** The proposed adversarial face synthesis method in Chapter 3, namely *AdvFaces*, can be utilized to launch adversarial attacks on other biometric systems such as automated fingerprint or iris recognition systems. For instance, by replacing the auxiliary AFR system with an automated fingerprint recognition system (such as DeepPrint [234]), we can synthesize adversarial fingerprints.

- **Robustness via Synthesis** Instead of utilizing a dedicated adversarial detector, we can utilize *FaceGuard* (Chapter 3) in an adversarial training mechanism. *FaceGuard*'s generator

can synthesize adversarial faces on-the-fly, while a face recognition system is trained to correctly classify the synthesized adversarial faces. In this manner, we should be able to obtain a face recognition system that is robust to adversarial faces.

- **Universal Attack Detection via Synthesis** Following the ideas presented in Chapter 3, we can also learn to synthesize physical spoofs along with adversarial attacks on-the-fly, such that a detector is concurrently trained to detect both synthesized spoofs and adversarial attacks. Once trained, the detector should be able to reliably reject face attacks from both physical and digital domains.

Chapter 6

PhD Overview

6.1 Publications

A list of all publications during the course of my PhD program (in reverse chronological order):

1. D. Deb, X. Liu and A. K. Jain, ”Unified Detection of Digital and Physical Face Attacks”, arXiv:2104.02156, 2021.

2. D. Deb, X. Liu and A. K. Jain, ”FaceGuard: A Self-Supervised Defense Against Adversarial Face Images”, arXiv:2011.14218, 2021.

3. D. Deb, D. Aggarwal and A. K. Jain. “Child Face Age-Progression via Deep Feature Aging”, *IEEE ICPR*, 2021.

4. J. J. Engelsma, D. Deb, K. Cao, A. Bhatnagar, P. S. Sudhish and A. K. Jain, ”Infant-ID: Fingerprints for Global Good”, in *IEEE PAMI*, 2021.

5. D. Deb and A. K. Jain. “Look Locally Infer Globally: A Generalizable Face Anti-Spoofing Approach”, in *IEEE TIFS*, 2020.

6. H. Xu, Y. Ma, H. Liu, D. Deb, H. Liu, J. Tang and A. K. Jain, “Adversarial Attacks and Defenses in Images, Graphs and Text: A Review”, in *IJAC*, DOI 10.1007/s11633-019-1211-x, 2020.

7. D. Deb, J. Zhang and A. K. Jain, “AdvFaces: Adversarial Face Synthesis”, in *IEEE IJCB*, 2020.

8. D. Deb, A. Ross, A. K. Jain, K. Prakah-Asante and K. Venkatesh Prasad, “Actions Speak Louder Than (Pass)words: Passive Authentication of Smartphone Users via Deep Temporal Features”, in *IEEE ICB*, 2019.

9. J. J. Engelsma, D. Deb, A. K. Jain, P. S. Sudhish and A. Bhatnagar, ”Infant-Prints: Fingerprints for Reducing Infant Mortality”, in *IEEE CVPRW-CV4GC*, 2019.

10. Y. Shi, D. Deb and A. K. Jain, “WarpGAN: Automatic Caricature Generation”, in *IEEE CVPR*, 2019.

11. D. Deb, N. Nain and A. K. Jain, “Longitudinal Study of Child Face Recognition”, in *IEEE ICB*, 2018.

12. D. Deb, S. Wiper, S. Gong, Y. Shi, C. Tymoszek, A. Fletcher and A. K. Jain, “Face Recognition: Primates in the Wild”, in *IEEE BTAS*, 2018.

13. E. Tabassi, T. Chugh, D. Deb and A. K. Jain, “Altered Fingerprints: Detection and Localization”, in *IEEE BTAS*, 2018.

14. D. Deb, T. Chugh, J. Engelsma, K. Cao, N. Nain, J. Kendall and A. K. Jain, “Matching Fingerphotos to Slap Fingerprint Images”, arXiv:1804.08122, 2018.

15. D. Deb, L. Best-Rowden and A. K. Jain, “Face Recognition Performance Under Aging”, in *IEEE CVPRW*, 2017.

6.2 Videos & Demos

The following videos demonstrate research solutions presented in the above publications in the real world:

1. Face Anti-spoofing https://youtu.be/VzD1GSJ5omQ

2. Adversarial Face Synthesis https://youtu.be/uZBKymweNvI

3. Automatic Caricature Synthesis https://youtu.be/zJL-eivtVnk

4. PrimID: Face Recognition for Endangered Primates https://youtu.be/mbiIhEjKfhA

6.3 Media Coverage

1. https://msutoday.msu.edu/news/2021/using-thumbprints-vaccination-records-to-save-lives (MSU Today)

2. https://www.sciencedaily.com/releases/2018/05/180524112345.htm (Science Daily)

3. https://msutoday.msu.edu/news/2018/msu-technology-and-app-could-help-endangered-primates-slow-illegal-trafficking (MSU Today)

4. https://www.springwise.com/app-identifies-endangered-primates-using-facial-recognition (Springwise)

5. https://www.conservationjobs.co.uk/articles/new-technology-assists-the-protection-of-primates (ConservationJobs)

6. https://www.asmag.com/rankings/m/content.aspx?id=25402 (ASMag)

7. https://olhardigital.com.br/en/2018/05/28/noticias/novo-sistema-de-reconhecimento-facial-pode-ajudar-a-salvar-primatas-da-extincao/ (OlharDigital)

8. https://freetheapes.org/tag/facial-recognition/ (PEGAS)

9. https://msutoday.msu.edu/news/2019/compact-low-cost-fingerprint-reader-could-help-reduce-infant-mortality-around-the-world (MSU Today)

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Y. Liu, J. Stehouwer, A. Jourabloo, and X. Liu, "Deep tree learning for zero-shot face anti-spoofing," in *CVPR*, 2019.

[2] Z. Boulkenafet, J. Komulainen, L. Li, X. Feng, and A. Hadid, "OULU-NPU: A mobile face presentation attack database with real-world variations," in *IEEE FG*, pp. 612–618, 2017.

[3] Z. Zhang, J. Yan, S. Liu, Z. Lei, D. Yi, and S. Z. Li, "A face antispoofing database with diverse attacks," in *IEEE ICB*, pp. 26–31, 2012.

[4] I. Chingovska, A. Anjos, and S. Marcel, "On the Effectiveness of Local Binary Patterns in Face Anti-spoofing," in *IEEE BIOSIG*, 2012.

[5] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, "Retinaface: Single-shot multi-level face localisation in the wild," in *IEEE CVPR*, pp. 5203–5212, 2020.

[6] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *IEEE CVPR*, pp. 815–823, 2015.

[7] BLCV, "Demystifying Face Recognition IV: Face-Alignment." https://bit.ly/3iUUBqz, 2017.

[8] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Tech. Rep. 07-49, University of Massachusetts, Amherst, October 2007.

[9] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *IEEE CVPR*, pp. 4690–4699, 2019.

[10] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014.

[11] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale.," *ICLR*, 2017.

[12] X. Liu and C.-J. Hsieh, "Rob-gan: Generator, discriminator, and adversarial attacker," in *CVPR*, 2019.

[13] Y. Jang, T. Zhao, S. Hong, and H. Lee, "Adversarial defense via learning to generate diverse attacks," in *ICCV*, 2019.

[14] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *ACM CCS*, pp. 135–147, 2017.

[15] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," *ICLR*, 2018.

[16] D. Deb, J. Zhang, and A. K. Jain, "Advfaces: Adversarial face synthesis," *arXiv preprint arXiv:1908.05008*, 2019.

[17] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *ICCV*, December 2015.

[18] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *CVPR*, 2019.

[19] D. Deb, X. Liu, and A. K. Jain, "Faceguard: A self-supervised defense against adversarial face images," *arXiv preprint arXiv:2011.14218*, 2020.

[20] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. K. Jain, "On the detection of digital face manipulation," in *CVPR*, 2020.

[21] D. Deb and A. K. Jain, "Look locally infer globally: A generalizable face anti-spoofing approach," *IEEE TIFS*, vol. 16, pp. 1143–1157, 2020.

[22] P. Grother, M. Ngan, and K. Hanaoka, "Ongoing face recognition vendor test (frvt)," *NIST Interagency Report*, 2018.

[23] S. Baker, T. Sim, and M. Bsat, "The cmu pose, illumination, and expression database," *IEEE TIFS*, vol. 25, no. 12, pp. 1615–1618, 2003.

[24] S. Sengupta, J.-C. Chen, C. Castillo, V. M. Patel, R. Chellappa, and D. W. Jacobs, "Frontal to profile face verification in the wild," in *IEEE WACV*, pp. 1–9, 2016.

[25] S. Moschoglou, A. Papaioannou, C. Sagonas, J. Deng, I. Kotsia, and S. Zafeiriou, "Agedb: the first manually collected, in-the-wild age database," in *IEEE CVPR*, pp. 51–59, 2017.

[26] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The feret evaluation methodology for face-recognition algorithms," *IEEE PAMI*, vol. 22, no. 10, pp. 1090–1104, 2000.

[27] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, and A. K. Jain, "Pushing the frontiers of unconstrained face detection and recognition: IARPA Janus Benchmark A," in *IEEE CVPR*, pp. 1931–1939, 2015.

[28] H. Goldstein, *Multilevel statistical models*, vol. 922. John Wiley & Sons, 2011.

[29] Z. Cheng, X. Zhu, and S. Gong, "Low-resolution face recognition," in *ACCV*, pp. 605–621, 2018.

[30] Y. Liu, A. Jourabloo, and X. Liu, "Learning deep models for face anti-spoofing: Binary or auxiliary supervision," in *IEEE CVPR*, June 2018.

[31] N. K. Ratha, J. H. Connell, and R. M. Bolle, "Enhancing security and privacy in biometrics-based authentication systems," *IBM Systems Journal*, vol. 40, no. 3, pp. 614–634, 2001.

[32] A. Dabouei, S. Soleymani, J. Dawson, and N. Nasrabadi, "Fast geometrically-perturbed adversarial faces," in *IEEE WACV*, pp. 1979–1988, 2019.

[33] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[34] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[35] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[36] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *NeurIPS*, 2017.

[37] L. Best-Rowden and A. K. Jain, "Longitudinal study of automatic face recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 1, pp. 148–162, 2017.

[38] D. Deb, N. Nain, and A. K. Jain, "Longitudinal study of child face recognition," in *IEEE ICB*, pp. 225–232, 2018.

[39] D. Deb, L. Best-Rowden, and A. K. Jain, "Face recognition performance under aging," in *IEEE CVPR Workshop*, pp. 46–54, 2017.

[40] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE CVPR*, vol. 1, pp. I–I, 2001.

[41] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE SPL*, vol. 23, no. 10, pp. 1499–1503, 2016.

[42] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," in *IEEE CVPR*, 2018.

[43] D. Wang, C. Otto, and A. K. Jain, "Face search at scale," *IEEE PAMI*, vol. 39, no. 6, pp. 1122–1136, 2016.

[44] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *IEEE CVPR*, pp. 212–220, 2017.

[45] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *IEEE CVPR*, pp. 586–587, 1991.

[46] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher vector faces in the wild.," in *BMVC*, vol. 2, p. 4, 2013.

[47] D. Chen, X. Cao, F. Wen, and J. Sun, "Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification," in *IEEE CVPR*, pp. 3025–3032, 2013.

[48] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: A joint formulation," in *ECCV*, pp. 566–579, 2012.

[49] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *IEEE CVPR*, pp. 1701–1708, 2014.

[50] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *IEEE CVPR*, pp. 1891–1898, 2014.

[51] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *BMVC*, pp. 41.1–41.12, 2015.

[52] H. T. F. Rhodes, *Alphonse Bertillon, father of scientific detection*. Greenwood, 1968.

[53] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *ECCV*, pp. 499–515, Springer, 2016.

[54] P. J. Phillips, P. J. Grother, R. J. Micheals, D. M. Blackburn, E. Tabassi, and M. Bone, "Face recognition vendor test 2002: Evaluation report," tech. rep., NIST, 2003.

[55] P. Flanagan, "Multiple biometric evaluation (mbe)," *NIST*, 2010.

[56] P. Grother and M. Ngan, "Face recognition vendor test (frvt): Performance of face identification algorithms," *NIST Interagency report*, vol. 8009, no. 5, p. 14, 2014.

[57] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *IEEE FG*, pp. 67–74, IEEE, 2018.

[58] S. Jia, G. Guo, and Z. Xu, "A survey on 3d mask presentation attack detection and countermeasures," *Pattern Recognition*, vol. 98, p. 107032, 2020.

[59] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting world leaders against deep fakes.," in *CVPR Workshops*, 2019.

[60] X. Yang, D. Yang, Y. Dong, W. Yu, H. Su, and J. Zhu, "Delving into the adversarial robustness on face recognition," *arXiv preprint arXiv:2007.04118*, 2020.

[61] A. George and S. Marcel, "Learning one class representations for face presentation attack detection using multi-channel convolutional neural networks," *IEEE TIFS*, vol. 16, pp. 361–375, 2020.

[62] Y. Liu, J. Stehouwer, and X. Liu, "On disentangling spoof trace for generic face anti-spoofing," in *ECCV*, Springer, 2020.

[63] H. Feng, Z. Hong, H. Yue, Y. Chen, K. Wang, J. Han, J. Liu, and E. Ding, "Learning generalized spoof cues for face anti-spoofing," *arXiv preprint arXiv:2005.03922*, 2020.

[64] International Standards Organization, "ISO/IEC 30107-1:2016, Information Technology Biometric Presentation Attack Detection Part 1: Framework." https://www.iso.org/standard/53227.html, 2016.

[65] S. Marcel, M. S. Nixon, and S. Z. Li, *Handbook of Biometric Anti-Spoofing*, vol. 1. Springer, 2014.

[66] I. Manjani, S. Tariyal, M. Vatsa, R. Singh, and A. Majumdar, "Detecting silicone mask-based presentation attack via deep dictionary learning," *IEEE TIFS*, vol. 12, no. 7, pp. 1713–1723, 2017.

[67] Y. Xu, T. Price, J.-M. Frahm, and F. Monrose, "Virtual u: Defeating face liveness detection by building virtual models from your public photos," in *USENIX*, pp. 497–512, 2016.

[68] "FastPass- a harmonized, modular reference system for all European automated border-crossing points." FastPass-EU, https://www.fastpass-project.eu.

[69] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[70] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[71] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *IEEE CVPR*, pp. 1765–1773, 2017.

[72] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *IEEE CVPR*, pp. 9185–9193, 2018.

[73] Wikipedia, "U.S. Customs and Border Protection." https://en.wikipedia.org/wiki/U.S._Customs_and_Border_Protection, 2019.

[74] U.S. Customs and Border Protection, "On a Typical Day in Fiscal Year 2018." https://www.cbp.gov/newsroom/stats/typical-day-fy2018, 2018.

[75] "Biometrics." U.S. Customs and Border Protection, https://www.cbp.gov/travel/biometrics.

[76] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *CVPR*, 2016.

[77] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," in *ICCV*, pp. 1–11, 2019.

[78] 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 *CVPR*, 2018.

[79] M. Liu, Y. Ding, M. Xia, X. Liu, E. Ding, W. Zuo, and S. Wen, "Stgan: A unified selective transfer network for arbitrary image attribute editing," in *CVPR*, 2019.

[80] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *CVPR*, 2020.

[81] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *IEEE CVPR*, pp. 2387–2395, 2016.

[82] Daily Mail, "Police arrest passenger who boarded plane in Hong Kong as an old man in flat cap and arrived in Canada a young Asian refugee." http://dailym.ai/2UBEcxO, 2011.

[83] The Verge, "This $150 mask beat Face ID on the iPhone X." https://bit.ly/300bRoC, 2017.

[84] N. Erdogmus and S. Marcel, "Spoofing in 2D Face Recognition with 3D Masks and Anti-spoofing with Kinect," in *IEEE BTAS*, 2013.

[85] D. Wen, H. Han, and A. K. Jain, "Face spoof detection with image distortion analysis," *IEEE TIFS*, vol. 10, no. 4, pp. 746–761, 2015.

[86] A. Costa-Pazo, S. Bhattacharjee, E. Vazquez-Fernandez, and S. Marcel, "The replay-mobile face presentation-attack database," in *IEEE BIOSIG*, Sept. 2016.

[87] S. Liu, B. Yang, P. C. Yuen, and G. Zhao, "A 3D Mask Face Anti-Spoofing Database with Real World Variations," in *IEEE CVPR*, 2016.

[88] Z. Yu, C. Zhao, Z. Wang, Y. Qin, Z. Su, X. Li, F. Zhou, and G. Zhao, "Searching central difference convolutional networks for face anti-spoofing," *arXiv preprint arXiv:2003.04092*, 2020.

[89] K. Kollreider, H. Fronthaler, M. I. Faraj, and J. Bigun, "Real-time face detection and motion analysis with application in "liveness" assessment," *IEEE TIFS*, vol. 2, no. 3, pp. 548–558, 2007.

[90] G. Pan, L. Sun, Z. Wu, and S. Lao, "Eyeblink-based anti-spoofing in face recognition from a generic webcamera," in *IEEE CVPR*, pp. 1–8, 2007.

[91] K. Patel, H. Han, and A. K. Jain, "Cross-database face antispoofing with robust feature representation," in *CCBR*, pp. 611–619, 2016.

[92] R. Shao, X. Lan, and P. C. Yuen, "Deep convolutional dynamic texture learning with adaptive channel-discriminability for 3d mask face anti-spoofing," in *IEEE IJCB*, pp. 748–755, 2017.

[93] J. Komulainen, H. Abdenour, and P. Matti, "Context based face anti-spoofing.," in *IEEE BTAS*, 2013.

[94] T. de Freitas Pereira, A. Anjos, J. M. De Martino, and S. Marcel, "LBP-TOP based countermeasure against face spoofing attacks," in *ACCV*, pp. 121–132, 2012.

[95] L. Li, Z. Xia, A. Hadid, X. Jiang, F. Roli, and X. Feng, "Face presentation attack detection in learned color-liked space," *arXiv preprint arXiv:1810.13170*, 2018.

[96] K. Patel, H. Han, and A. K. Jain, "Secure face unlock: Spoof detection on smartphones," *IEEE TIFS*, vol. 11, no. 10, pp. 2268–2283, 2016.

[97] J. Galbally, S. Marcel, and J. Fierrez, "Image quality assessment for fake biometric detection: Application to iris, fingerprint, and face recognition," *IEEE TIP*, vol. 23, no. 2, pp. 710–724, 2014.

[98] H. Li, S. Wang, and A. C. Kot, "Face spoofing detection with image quality regression," in *IEEE IPTA*, pp. 1–6, 2016.

[99] J. Li, Y. Wang, T. Tan, and A. K. Jain, "Live face detection based on the analysis of fourier spectra," in *Biometric Technology for Human Identification*, vol. 5404, pp. 296–303, SPIE, 2004.

[100] T. Pereira, A. Anjos, J. M. De Martino, and S. Marcel, "Can face anti-spoofing countermeasures work in a real world scenario?," in *IEEE ICB*, 2013.

[101] J. Määttä, A. Hadid, and M. Pietikäinen, "Face spoofing detection from single images using micro-texture analysis," in *IEEE IJCB*, pp. 1–7, 2011.

[102] Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face spoofing detection using colour texture analysis," *IEEE TIFS*, vol. 11, no. 8, pp. 1818–1830, 2016.

[103] J. Yang, Z. Lei, S. Liao, and S. Z. Li, "Face liveness detection with component dependent descriptor," in *IEEE ICB*, pp. 1–6, 2013.

[104] X. Tan, Y. Li, J. Liu, and L. Jiang, "Face liveness detection from a single image with sparse low rank bilinear discriminative model," in *ECCV*, pp. 504–517, 2010.

[105] Z. Boulkenafet, J. Komulainen, and A. Hadid, "Face antispoofing using speeded-up robust features and fisher vector encoding," *IEEE Signal Processing Letters*, vol. 24, no. 2, pp. 141–145, 2017.

[106] T. Wang, J. Yang, Z. Lei, S. Liao, and S. Z. Li, "Face liveness detection using 3d structure recovered from a single camera," in *IEEE ICB*, 2013.

[107] Y. Wang, F. Nian, T. Li, Z. Meng, and K. Wang, "Robust face anti-spoofing with depth information," *Journal of Visual Communication and Image Representation*, vol. 49, 09 2017.

[108] S. Zhang, X. Wang, A. Liu, C. Zhao, J. Wan, S. Escalera, H. Shi, Z. Wang, and S. Z. Li, "CASIA-SURF: A Dataset and Benchmark for Large-scale Multi-modal Face Anti-Spoofing," *arXiv preprint arXiv:1812.00408*, 2018.

[109] V. Conotter, E. Bodnari, G. Boato, and H. Farid, "Physiologically-based detection of computer generated faces in video," *IEEE ICIP*, pp. 248–252, 01 2015.

[110] Z. Zhang, D. Yi, Z. Lei, and S. Li, "Face liveness detection by learning multispectral reflectance distributions," in *IEEE FG*, pp. 436 – 441, 2011.

[111] G. Chetty, "Biometric liveness checking using multimodal fuzzy fusion," in *IEEE WCCI*, pp. 1–8, 07 2010.

[112] J. Yang, Z. Lei, and S. Z. Li, "Learn Convolutional Neural Network for Face Anti-Spoofing," *arXiv preprint arXiv:1408.5601*, 2014.

[113] N. N. Lakshminarayana, N. Narayan, N. Napp, S. Setlur, and V. Govindaraju, "A discriminative spatio-temporal mapping of face for liveness detection," in *IEEE ISBA*, pp. 1–7, 2017.

[114] X. Tu and Y. Fang, "Ultra-deep neural network for face anti-spoofing," in *NIPS*, pp. 686–695, 2017.

[115] O. Lucena, A. Junior, V. Moia, R. Souza, E. Valle, and R. Lotufo, "Transfer learning using convolutional neural networks for face anti-spoofing," in *ICIAR*, pp. 27–34, 2017.

[116] L. Li, X. Feng, Z. Boulkenafet, Z. Xia, M. Li, and A. Hadid, "An original face anti-spoofing approach using partial convolutional neural network," in *IEEE IPTA*, pp. 1–6, 2016.

[117] S. R. Arashloo, J. Kittler, and W. Christmas, "An anomaly detection approach to face spoofing detection: A new formulation and evaluation protocol," *IEEE Access*, vol. 5, pp. 13868–13882, 2017.

[118] O. Nikisins, A. Mohammadi, A. Anjos, and S. Marcel, "On effectiveness of anomaly detection approaches against unseen presentation attacks in face anti-spoofing," in *IEEE ICB*, pp. 75–81, 2018.

[119] D. Pérez-Cabo, D. Jiménez-Cabello, A. Costa-Pazo, and R. J. López-Sastre, "Deep anomaly detection for generalized face anti-spoofing," in *IEEE CVPR*, pp. 0–0, 2019.

[120] H. Li, W. Li, H. Cao, S. Wang, F. Huang, and A. C. Kot, "Unsupervised domain adaptation for face anti-spoofing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1794–1809, 2018.

[121] S. R. Arashloo, J. Kittler, and W. Christmas, "Face spoofing detection based on multiple descriptor fusion using multiscale dynamic binarized statistical image features," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 11, pp. 2396–2407, 2015.

[122] J. Yang, Z. Lei, D. Yi, and S. Z. Li, "Person-specific face antispoofing with subject domain adaptation," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 797–809, 2015.

[123] Y. Atoum, Y. Liu, A. Jourabloo, and X. Liu, "Face anti-spoofing using patch and depth-based cnns," in *2017 IEEE IJCB*, pp. 319–328, 2017.

[124] A. George and S. Marcel, "Deep pixel-wise binary supervision for face presentation attack detection," *arXiv preprint arXiv:1907.04047*, 2019.

[125] A. Jourabloo, Y. Liu, and X. Liu, "Face de-spoofing: Anti-spoofing via noise modeling," in *ECCV*, pp. 290–306, 2018.

[126] C. Nagpal and S. R. Dubey, "A performance evaluation of convolutional neural networks for face anti spoofing," in *IEEE IJCNN*, pp. 1–8, 2019.

[127] W. Sun, Y. Song, C. Chen, J. Huang, and A. C. Kot, "Face spoofing detection based on local ternary label supervision in fully convolutional networks," *IEEE TIFS*, vol. 15, pp. 3181–3196, 2020.

[128] W. Sun, Y. Song, H. Zhao, and Z. Jin, "A face spoofing detection method based on domain adaptation and lossless size adaptation," *IEEE Access*, vol. 8, pp. 66553–66563, 2020.

[129] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[130] D. E. King, "Dlib-ml: A machine learning toolkit," *JMLR*, vol. 10, no. Jul, pp. 1755–1758, 2009.

[131] Z. Boulkenafet, J. Komulainen, Z. Akhtar, A. Benlamoudi, D. Samai, S. E. Bekhouche, A. Ouafi, F. Dornaika, A. Taleb-Ahmed, L. Qin, *et al.*, "A competition on generalized software-based face presentation attack detection in mobile scenarios," in *IEEE IJCB*, pp. 688–696, 2017.

[132] H. Chen, G. Hu, Z. Lei, Y. Chen, N. M. Robertson, and S. Z. Li, "Attention-based two-stream convolutional networks for face spoofing detection," *IEEE TIFS*, vol. 15, pp. 578–593, 2019.

[133] R. Bresan, A. Pinto, A. Rocha, C. Beluzo, and T. Carvalho, "Facespoof buster: a presentation attack detector based on intrinsic image properties and deep learning," *arXiv preprint arXiv:1902.02845*, 2019.

[134] N. Damer, K. Dimitrov, R. Wilson, E. Hancock, and W. Smith, "Practical view on face presentation attack detection.," in *BMVC*, 2016.

[135] X. Yang, W. Luo, L. Bao, Y. Gao, D. Gong, S. Zheng, Z. Li, and W. Liu, "Face anti-spoofing: Model matters, so does data," in *IEEE CVPR*, pp. 3507–3516, 2019.

[136] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *IEEE SP*, pp. 39–57, 2017.

[137] C. Xiao, J.-Y. Zhu, B. Li, W. He, M. Liu, and D. Song, "Spatially transformed adversarial examples," *arXiv preprint arXiv:1801.02612*, 2018.

[138] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning models," *arXiv preprint arXiv:1707.08945*, 2017.

[139] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *IEEE EuroS&P*, pp. 372–387, 2016.

[140] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[141] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *IEEE CVPR*, pp. 2574–2582, 2016.

[142] Y. Dong, H. Su, B. Wu, Z. Li, W. Liu, T. Zhang, and J. Zhu, "Efficient decision-based black-box adversarial attacks on face recognition," in *IEEE CVPR*, pp. 7714–7722, 2019.

[143] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," *arXiv preprint arXiv:1611.02770*, 2016.

[144] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, pp. 2672–2680, 2014.

[145] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[146] E. L. Denton, S. Chintala, and R. Fergus, "Deep generative image models using a laplacian pyramid of adversarial networks," in *NIPS*, pp. 1486–1494, 2015.

[147] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. S. Lempitsky, "Texture networks: Feed-forward synthesis of textures and stylized images.," in *ICML*, vol. 1, p. 4, 2016.

[148] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV*, pp. 694–711, 2016.

[149] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *IEEE CVPR*, pp. 2414–2423, 2016.

[150] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE CVPR*, pp. 1125–1134, 2017.

[151] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE ICCV*, pp. 2223–2232, 2017.

[152] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *NIPS*, pp. 2234–2242, 2016.

[153] M. F. Mathieu, J. J. Zhao, J. Zhao, A. Ramesh, P. Sprechmann, and Y. LeCun, "Disentangling factors of variation in deep representation using adversarial training," in *NIPS*, pp. 5040–5048, 2016.

[154] S. Baluja and I. Fischer, "Adversarial transformation networks: Learning to generate adversarial examples," *arXiv preprint arXiv:1703.09387*, 2017.

[155] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," *arXiv preprint arXiv:1801.02610*, 2018.

[156] X. Wang, K. He, C. Guo, K. Q. Weinberger, and J. E. Hopcroft, "AT-GAN: A Generative Attack Model for Adversarial Transferring on Generative Adversarial Nets," *arXiv preprint arXiv:1904.07793*, 2019.

[157] Y. Song, R. Shu, N. Kushman, and S. Ermon, "Constructing unrestricted adversarial examples with generative models," in *NIPS*, pp. 8312–8323, 2018.

[158] A. J. Bose and P. Aarabi, "Adversarial attacks on face detectors using neural net based constrained optimization," in *IEEE MMSP*, pp. 1–6, 2018.

[159] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *ACM SIGSAC*, pp. 1528–1540, 2016.

[160] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "A general framework for adversarial examples with objectives," *ACM TOPS*, vol. 22, no. 3, p. 16, 2019.

[161] Q. Song, Y. Wu, and L. Yang, "Attacks on state-of-the-art face recognition using attentional adversarial attack generative network," *arXiv preprint arXiv:1811.12026*, 2018.

[162] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, IEEE, 2009.

[163] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," *Citeseer*, 2009.

[164] C. Xie, Y. Wu, L. v. d. Maaten, A. L. Yuille, and K. He, "Feature denoising for improving adversarial robustness," in *CVPR*, 2019.

[165] Y. LeCun, "The mnist database of handwritten digits," *Tech Report*, 1998.

[166] Z. Gong, W. Wang, and W.-S. Ku, "Adversarial and clean data are not twins," *arXiv preprint arXiv:1704.04960*, 2017.

[167] A. Agarwal, R. Singh, M. Vatsa, and N. Ratha, "Are image-agnostic universal adversarial perturbations for face recognition difficult to detect?," in *BTAS*, 2018.

[168] A. P. Founds, N. Orlans, W. Genevieve, and C. I. Watson, "NIST special database 32-multiple encounter dataset II (MEDS-II).," in *NIST Intragency Report*, 2011.

[169] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-PIE.," in *FG*, 2010.

[170] J. R. Beveridge, P. J. Phillips, D. S. Bolme, B. A. Draper, G. H. Givens, Y. M. Lui, M. N. Teli, H. Zhang, W. T. Scruggs, K. W. Bowyer, P. J. Flynn, and S. Cheng, "The challenge of face recognition from digital point-and-shoot cameras.," in *BTAS*, 2013.

[171] Moosavi-Dezfooli, Seyed-Mohsen, A. Fawzi, O. Fawzi, and P. Frossard, "From few to many: Illumination cone models for face recognition under variable lighting and pose.," in *CVPR*, pp. 1765–1773, 2017.

[172] A. Goel, A. Singh, A. Agarwal, M. Vatsa, and R. Singh, "Smartbox: Benchmarking adversarial detection and mitigation algorithms for face recognition.," in *BTAS*, pp. 1–7, 2018.

[173] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose.," in *PAMI*, pp. 643–660, 2001.

[174] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "EAD: Elastic-net attacks to deep neural networks via adversarial examples.," *AAAI*, 2018.

[175] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," *ICLR*, 2018.

[176] G. Goswami, A. Agarwal, N. Ratha, R. Singh, and M. Vatsa, "Detecting and mitigating adversarial perturbations for robust face recognition," *ICCV*, vol. 127, no. 6-7, pp. 719–742, 2019.

[177] P. J. Phillips, P. J. Flynn, J. R. Beveridge, W. T. Scruggs, A. J. O'toole, D. Bolme, K. W. Bowyer, B. A. Draper, G. H. Givens, Y. M. Lui, *et al.*, "Overview of the multiple biometrics grand challenge," in *ICB*, 2010.

[178] J. Liu, W. Zhang, Y. Zhang, D. Hou, Y. Liu, H. Zha, and N. Yu, "Detection based defense against adversarial examples from the steganalysis point of view," in *CVPR*, 2019.

[179] F. V. Massoli, F. Carrara, G. Amato, and F. Falchi, "Detection of face recognition adversarial attacks," *CVIU*, p. 103103, 2020.

[180] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," in *FG*, 2018.

[181] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world.," *arXiv preprint arXiv:1607.02533*, 2016.

[182] A. Agarwal, R. Singh, M. Vatsa, and N. K. Ratha, "Image transformation based defense against adversarial perturbation on deep learning models," *IEEE Transactions on Dependable and Secure Computing*, 2020.

[183] Z. Liu, Q. Liu, T. Liu, N. Xu, X. Lin, Y. Wang, and W. Wen, "Feature distillation: Dnn-oriented jpeg compression against adversarial examples," in *CVPR*, 2019.

[184] M. Naseer, S. Khan, M. Hayat, F. S. Khan, and F. Porikli, "A self-supervised approach for adversarial robustness," in *CVPR*, 2020.

[185] J. Zhou, C. Liang, and J. Chen, "Manifold projection for adversarial defense on face recognition," in *European Conference on Computer Vision*, pp. 288–305, Springer, 2020.

[186] H. Qiu, C. Xiao, L. Yang, X. Yan, H. Lee, and B. Li, "Semanticadv: Generating adversarial examples via attribute-conditional image editing," *arXiv preprint arXiv:1906.07927*, 2019.

[187] D. Su, H. Zhang, H. Chen, J. Yi, P.-Y. Chen, and Y. Gao, "Is robustness the cost of accuracy?–a comprehensive study on the robustness of 18 deep image classification models.," in *ECCV*, 2018.

[188] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy.," *ICLR*, 2017.

[189] G. S. Dhillon, K. Azizzadenesheli, Z. C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar, "Stochastic activation pruning for robust adversarial defense," in *ICLR*, 2018.

[190] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting adversarial samples from artifacts," *arXiv preprint arXiv:1703.00410*, 2017.

[191] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," *arXiv preprint arXiv:1702.06280*, 2017.

[192] X. Li and F. Li, "Adversarial examples detection in deep networks with convolutional filter statistics," in *ICCV*, pp. 5764–5772, 2017.

[193] D. Hendrycks and K. Gimpel, "Early methods for detecting adversarial images," *arXiv preprint arXiv:1608.00530*, 2016.

[194] C. Guo, M. Rana, M. Cisse, and L. Van Der Maaten, "Countering adversarial images using input transformations," *arXiv preprint arXiv:1711.00117*, 2017.

[195] H. Kannan, A. Kurakin, and I. Goodfellow, "Adversarial logit pairing," *arXiv preprint arXiv:1803.06373*, 2018.

[196] J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff, "On detecting adversarial perturbations," *ICLR*, 2017.

[197] T. Na, J. H. Ko, and S. Mukhopadhyay, "Cascade adversarial machine learning regularized with a unified embedding," *ICLR*, 2017.

[198] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," *ICLR*, 2017.

[199] V. Zantedeschi, M.-I. Nicolae, and A. Rawat, "Efficient defenses against adversarial attacks," in *ACM Workshop on Artificial Intelligence and Security*, pp. 39–49, 2017.

[200] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *ACM Workshop on Artificial Intelligence and Security*, pp. 3–14, 2017.

[201] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples," *ICML*, 2018.

[202] N. Carlini and D. Wagner, "Magnet and "efficient defenses against adversarial attacks" are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478*, 2017.

[203] M. Mosbach, M. Andriushchenko, T. Trost, M. Hein, and D. Klakow, "Logit pairing methods can fool gradient-based attacks," *arXiv preprint arXiv:1810.12042*, 2018.

[204] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," *ICLR*, 2017.

[205] A. Rozsa, M. Günther, and T. E. Boult, "Lots about attacking deep features," in *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pp. 168–176, IEEE, 2017.

[206] G. Goswami, N. Ratha, A. Agarwal, R. Singh, and M. Vatsa, "Unravelling robustness of deep learning based face recognition against adversarial attacks," in *AAAI*, 2018.

[207] P. J. Grother, M. Ngan, and K. Hanaoka, "Ongoing Face Recognition Vendor Test (FRVT), Part 2: Identification," *NIST Interagency Report*, 2018.

[208] Y. Liu, A. Jourabloo, and X. Liu, "Learning deep models for face anti-spoofing: Binary or auxiliary supervision," in *CVPR*, 2018.

[209] Y. Liu, J. Stehouwer, and X. Liu, "On disentangling spoof traces for generic face anti-spoofing," in *ECCV*, 2020.

[210] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. Jain, "On the detection of digital face manipulation," in *CVPR*, 2020.

[211] S. Shan, E. Wenger, J. Zhang, H. Li, H. Zheng, and B. Y. Zhao, "Fawkes: protecting privacy against unauthorized deep learning models," in *USENIX*, pp. 1589–1604, 2020.

[212] A. Raghunathan, S. M. Xie, F. Yang, J. C. Duchi, and P. Liang, "Adversarial training can hurt generalization," *arXiv preprint arXiv:1906.06032*, 2019.

[213] D. Yang, S. Hong, Y. Jang, T. Zhao, and H. Lee, "Diversity-sensitive conditional generative adversarial networks," *ICLR*, 2019.

[214] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Two-stream neural networks for tampered face detection," in *CVPRW*, IEEE, 2017.

[215] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *ICASSP*, IEEE, 2019.

[216] P. Korshunov and S. Marcel, "Deepfakes: a new threat to face recognition? assessment and detection," *arXiv preprint arXiv:1812.08685*, 2018.

[217] R. Wang, F. Juefei-Xu, L. Ma, X. Xie, Y. Huang, J. Wang, and Y. Liu, "Fakespotter: A simple yet robust baseline for spotting ai-synthesized fake faces," *arXiv preprint arXiv:1909.06122*, 2019.

[218] Y. Liu, A. Jourabloo, and X. Liu, "Learning deep models for face anti-spoofing: Binary or auxiliary supervision," in *CVPR*, June 2018.

[219] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv preprint arXiv:1706.06083*, 2017.

[220] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo, "Face x-ray for more general face forgery detection," in *CVPR*, 2020.

[221] Y. A. U. Rehman, L. M. Po, and M. Liu, "Deep learning for face anti-spoofing: an end-to-end approach," in *IEEE SPA*, pp. 195–200, 2017.

[222] A. Jourabloo, Y. Liu, and X. Liu, "Face de-spoofing: Anti-spoofing via noise modeling," in *ECCV*, 2018.

[223] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *CVPR*, 2017.

[224] S. Mehta, A. Uberoi, A. Agarwal, M. Vatsa, and R. Singh, "Crafting a panoptic face presentation attack detector," in *ICB*, IEEE, 2019.

[225] J. Stehouwer, A. Jourabloo, Y. Liu, and X. Liu, "Noise modeling, synthesis and classification for generic object anti-spoofing," in *CVPR*, 2020.

[226] M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning," *arXiv preprint arXiv:1511.06114*, 2015.

[227] E. Meyerson and R. Miikkulainen, "Pseudo-task augmentation: From deep multitask learning to intratask sharing—and back," in *ICML*, pp. 3511–3520, 2018.

[228] X. Yin and X. Liu, "Multi-task convolutional neural network for pose-invariant face recognition," *IEEE T-IP*, vol. 27, no. 2, pp. 964–975, 2017.

[229] M. Crawshaw, "Multi-task learning with deep neural networks: A survey," *arXiv preprint arXiv:2009.09796*, 2020.

[230] T. Gui, L. Qing, Q. Zhang, J. Ye, H. Yan, Z. Fei, and X. Huang, "Constructing multiple tasks for augmentation: Improving neural image classification with k-means features," in *AAAI*, 2020.

[231] K. Hsu, S. Levine, and C. Finn, "Unsupervised learning via meta-learning," *ICLR*, 2018.

[232] J. Baxter, "A bayesian/information theoretic model of learning to learn via multiple task sampling," *Machine learning*, vol. 28, no. 1, pp. 7–39, 1997.

[233] N. Sanghvi, S. K. Singh, A. Agarwal, M. Vatsa, and R. Singh, "Mixnet for generalized face presentation attack detection," *arXiv preprint arXiv:2010.13246*, 2020.

[234] J. J. Engelsma, K. Cao, and A. K. Jain, "Learning a fixed-length fingerprint representation," *IEEE PAMI*, 2019.