An Analysis on Adversarial Machine Learning: Methods and Applications

Ali Dabouei
ad0046@mix.wvu.edu

An Analysis on Adversarial Machine Learning: Methods and Applications

Ali Dabouei

Dissertation submitted to the
Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University

in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Electrical Engineering

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2022

Keywords: Adversarial Learning, Deep Learning, Adversarial Example, Adversarial Defense, Generative Adversarial Networks, Data Augmentation, Latent Fingerprint Enhancement, Biometrics.

Abstract

An Analysis on Adversarial Machine Learning: Methods and Applications

Ali Dabouei

Deep learning has witnessed astonishing advancement in the last decade and revolutionized many fields ranging from computer vision to natural language processing. A prominent field of research that enabled such achievements is adversarial learning, investigating the behavior and functionality of a learning model in presence of an adversary. Adversarial learning consists of two major trends. The first trend analyzes the susceptibility of machine learning models to manipulation in the decision-making process and aims to improve the robustness to such manipulations. The second trend exploits adversarial games between components of the model to enhance the learning process. This dissertation aims to provide an analysis on these two sides of adversarial learning and harness their potential for improving the robustness and generalization of deep models.

In the first part of the dissertation, we study the adversarial susceptibility of deep learning models. We provide an empirical analysis on the extent of vulnerability by proposing two adversarial attacks that explore the geometric and frequency-domain characteristics of inputs to manipulate deep decisions. Afterward, we formalize the susceptibility of deep networks using the first-order approximation of the predictions and extend the theory to the ensemble classification scheme. Inspired by theoretical findings, we formalize a reliable and practical defense against adversarial examples to robustify ensembles. We extend this part by investigating the shortcomings of adversarial training (AT) and highlight that the popular momentum stochastic gradient descent, developed essentially for natural training, is not proper for optimization in adversarial training since it is not designed to be robust against the chaotic behavior of gradients in this setup. Motivated by these observations, we develop an optimization method that is more suitable for adversarial training. In the second part of the dissertation, we harness adversarial learning to enhance the generalization and performance of deep networks in discriminative and generative tasks. We develop several models for biometric identification including fingerprint distortion rectification and latent fingerprint reconstruction. In particular, we develop a ridge reconstruction model based on generative adversarial networks that estimates the missing ridge information in latent fingerprints. We introduce a novel modification that enables the generator network to preserve the ID information during the reconstruction process. To address the scarcity of data, *e.g.*, in latent fingerprint analysis, we develop a supervised augmentation technique that combines input examples based on their salient regions. Our findings advocate that adversarial learning improves the performance and reliability of deep networks in a wide range of applications.

To my mother
in whom I saw the embodiment of love, selflessness, and dedication.

Acknowledgement

The last several years have been an exciting chapter in my life and provided me with a great opportunity to enhance my limited knowledge and develop personal and professional characteristics. I have been surrounded by distinguished individuals who have played an influential role in this chapter, and I would like to thank them and show my sincere appreciations.

I would like to express my warmest gratitude to my mentor and advisor Dr. Nasser M. Nasrabadi, who made this work possible. I thank him for welcoming me in his research lab, his impeccable guidance, inspirational perspective, enthusiastic attitude, and dedication to novel research through my doctorate program at West Virginia University. His compassionate and responsible character has guided me through the overwhelming obstacles in this journey, and this dissertation is undoubtedly resulted from his supervision. I also would like to thank Dr. Jeremy Dawson, Dr. Xin Li, Dr. Omid Dehzangi, and Dr. Matthew C. Valenti for serving on my committee and for their constructive suggestions and advice that refined this work.

This dissertation was supported by the fellowship award from the department of justice (DOJ grant 2019-R2-CX-0041). I thank Cathy Girouard for her excellent support and help for managing the grant.

Next, I would like to thank all past and current members of our research lab with whom I have had the pleasure of working. Particularly, I would like to thank Sobhan, Mehdi, Hadi, Fariborz, Domenick, Amirsina, Uche, Moktari, Amol, Poorya, Salman, Saba, Arash, Nyma, and Paria. I am grateful to all my colleagues for making the lab such a productive, lively, and dynamic environment.

Most importantly, my heartfelt thanks go to my family, my mother Fatemeh, and my affectionate sister Hamdam, who has been my oldest friend. I specifically thank my beautiful, whip-smart, and hilarious soulmate Sadaf for accompanying me throughout this journey from the very first moment. Without the love, support, and patience of them, none of my achievements would have been possible. To them I dedicate this dissertation.

Table of Contents

List of Tables

List of Figures

Acronyms

AT adversarial training. 2, 70–81, 83–88

DNN deep neural network. 1–3, 5, 6, 25, 27, 28, 30, 32, 46–49, 51, 70, 90, 109

ENGM example-normalized stochastic gradient descent with momentum. 4, 73, 77–79, 81–88

FGSM fast gradient sign method. 9–11, 14, 23

FLM fast landmark manipulation method. 18, 22, 23

FR face recognition. 3, 4, 7–9, 11–14, 17–20, 23, 89–93, 97, 99

GAL gradient alignment loss. 50, 54, 58, 62, 65, 67, 68

GAN Generative Adversarial Network. 2, 173

GFLM grouped fast landmark manipulation method. 18, 22, 23

GPMR joint gradient phase and magnitude regularization. 56–58, 62, 64, 65, 67, 68

MSGD momentum stochastic gradient descent. 4, 72–82, 84–88

NT natural training. 71–75, 78, 79, 87

ROC receiver operating characteristic. 68

SNGM normalized gradient descent with momentum. 75–77, 81

SOTA state-of-the-art. 3, 6, 8, 11, 17, 23, 24, 26, 27, 40, 47, 71, 91, 93, 129

TPS thin-plate spline. 18, 94

TV total variation. 28, 114

Chapter 1

Introduction

1.1 Problem and Motivation

Machine learning has witnessed astonishing advancement in the last decade and revolutionized many domains ranging from computer vision [12, 18] to natural language processing [19–21]. These achievements have been fueled by major progress in other domains that provided two types of resources for training deep neural network (DNN). First, the advancement of hardware technology has led to the development of graphics processing units with a large amounts of memory to store numerous training parameters and many processing cores to optimize these parameters efficiently via parallel processing. Second, the huge amount of data has become available which has enabled us to gather large training datasets and utilize them to enhance the generalization of the learning process. Consequently, the theory of DNNs has made significant progress in various directions including network architectures [12, 18, 22, 23], optimization techniques [24–26], learning paradigms [27, 28], and objective functions [29–31]. Despite the major progress in the field, there is still a long journey to understand and exploit the mysterious potential of learning systems.

One major trend of research that significantly contributed to the theory of learning systems is adversarial learning. Adversarial learning aims to analyze the functionality and behavior of learning models in presence of an adversary. Adversarial learning broadly encompasses two trends, based on the intention of the adversary. The first trend is concerned with the security of predictions, and the adversary is a distinct external entity who aims to alter

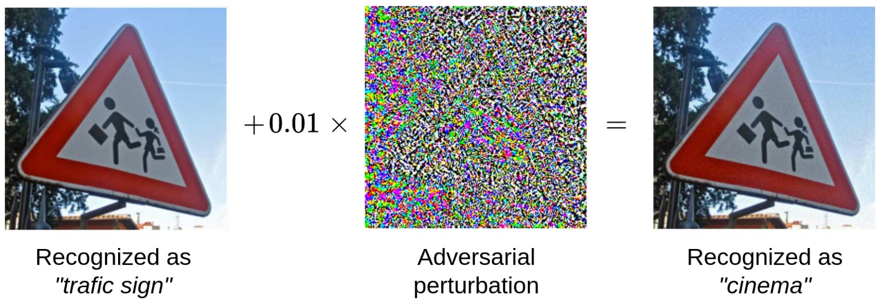


Figure 1.1: Illustration of an adversarial perturbation and its functionality. The original image on the left is correctly classified by the deep model as *"traffic sign"*. However, adding the adversarial perturbation on the middle changes the prediction of the model to *"cinema"*. Note that the perturbation on the output image is almost imperceptible to the human eye. Figure is inspired by Figure 1 in [10].

the learning or prediction process in order to change the decisions of the model [4, 10, 32, 33]. A well-known example for this kind of adversary is *adversarial example*: a manipulated version of the input data that remains benign to the human perception but can fool deep learning models to confidently make wrong predictions [10, 32]. Figure 1.1 provides a visualization of adversarial examples where an imperceptible additive perturbation changes the prediction of a deep model, *i.e.*, ResNet-50. Adversarial examples are potential threats to almost all applications of machine learning [34–36] and raise critical concerns regarding the deployment of machine learning models in security-sensitive applications such as autonomous vehicles [37] and biometric identification [4, 38]. While analyzing DNNs as differentiable transfer functions has led to substantial studies exploring embedding spaces and their characteristics in regard to training paradigms, analyzing the adversarial behavior has highlighted the importance of the topology of the decision boundaries and their properties in high dimensional data spaces [39, 40].

The second trend of adversarial learning utilizes an internal adversary as part of the model to improve the learning process. This goal is often obtained by designing a min-max game between the components of the model in which one component tries to maximize the loss function and the other component minimizes the loss simultaneously. A well-known example for this type of adversarial learning is Generative Adversarial Network (GAN)s [27, 28, 41], in which the target model itself is an adversary for the discriminator network.

The dissertation is broadly divided into two parts. The first part aims to analyze the adversarial susceptibility of DNNs to novel adversarial attacks and proposes approaches for improving the robustness of predictions against such manipulations of the input samples. In the second part of the dissertation, we harness adversarial learning and develop applications for computer vision and biometric identification. In the following sections, we provide an introduction to our contributions and the organization of the dissertation.

1.2 Adversarial Robustness

In the first part of the dissertation including Chapters 2, 3, 4, and 5, we analyze the susceptibility of DNNs to adversarial perturbations and propose approaches for alleviating this critical limitation of DNNs. In Chapter 2, we analyze the susceptibility of deep face recognition (FR) models to geometric manipulations of the face and propose adversarial attacks that modify the landmark locations of the face to compute adversarial examples. By reducing the search space for finding the adversarial manipulations, the proposed attack achieves notable speed-up compared to the previous geometric attacks, while maintaining a high success rate in fooling deep models. This method alters each landmark of the face independently, and therefore, sometimes causes artifacts in the generated faces. To address this shortcoming, we developed a second attack constrained on the semantic structure of the face. The second attack is extremely powerful in generating natural-looking adversarial faces that are hard to detect by the human eye and state-of-the-art (SOTA) defense methods.

In Chapter 3, the frequency characteristics of additive adversarial perturbation is studied and an adversarial attack capable of crafting perturbations with controllable frequency components is proposed. In Chapter 4, we exploit first-order interactions within ensembles to formalize a reliable and practical adversarial defense. We introduce a scenario of interactions that certifiably improves the robustness according to the size of the ensemble, the diversity of the gradient directions, and the balance of the member's contribution to the robustness.

One of the most effective methods (defenses) to alleviate adversarial susceptibility is adversarial training which improves the robustness by training the model on the worst-case loss [42, 43]. Despite the fundamental distinction between adversarial and natural training, adversarial training methods generally adopt momentum stochastic gradient descent (MSGD) for the outer optimization. In Chapter 5, we analyze this choice by investigating the overlooked role of outer optimization in adversarial training. Our exploratory evaluations reveal that adversarial training induces higher gradient norm and variance compared to natural training. This phenomenon hinders the outer optimization since the convergence rate of MSGD is highly dependent on the variance of the gradients. To this end, we propose an optimization method called example-normalized stochastic gradient descent with momentum (ENGM) which regularizes the contribution of each input example to the average mini-batch gradients. We prove that the convergence rate of ENGM is independent of the variance of the gradients, and thus, it is suitable for adversarial training. Furthermore, we introduce a trick to reduce the computational cost of ENGM using empirical observations on the correlation between the norm of gradients w.r.t. the network parameters and input examples. Through extensive evaluations and ablation studies we demonstrate that ENGM and its variants consistently improve the performance of adversarial training and alleviate its major shortcomings including robust overfitting and high sensitivity to hyperparameter settings.

1.3 Applications of Adversarial Learning

In the second part of the dissertation, we develop applications of machine learning and especially adversarial learning for problems in computer vision and biometrics. In Chapter 6, we propose a framework for disentangling deep representations for the two major characteristics of the face, namely appearance and geometry. To provide supervision for this aim, we generate geometrically identical faces by incorporating spatial transformations. We demonstrate that the proposed approach enhances the performance of deep FR models by assisting the training process in two ways. First, it enforces the early and intermediate convolutional layers to learn more representative features that satisfy the properties of disentangled embeddings. Second, it augments the training set by altering the geometry of the faces.

In Chapter 7, we study the mixing augmentation, a prominent approach for improving the generalization of DNNs. We demonstrate that the blind mixing in previous approaches such as MixUp [44,45] and CutMix [46] can potentially deteriorate the information in salient features of the input images. Consequently, we develop a mixing augmentation method that identifies the salient regions within input images to construct mixed training samples. Hence, SuperMix constructs mixed images rich in visual features and complying with realistic image priors. To enhance the efficiency of the optimization for SuperMix, a variant of the Newton iterative method, 65$\times$ faster than gradient descent is developed. Through extensive evaluations and ablation studies on two tasks of object classification and knowledge distillation, we demonstrate that SuperMix can significantly alleviate overfitting in DNNs.

In Chapter 8, we address the geometric distortion of fingerprints and their negative effect on the recognition performance by developing a fast and effective distortion estimator which captures the nonlinear properties of geometric distortion. While recently proposed methods handle distortion using a dictionary of distorted templates, we use DNNs to estimate the principal distortion components of input samples. Our approach does not require the ridge frequency and orientation maps to estimate the distortions. In addition, our model estimates the distortion parameters continuously to achieve more accurate rectifications.

In Chapter 9, we developed an adversarial learning model to reconstruct the ridge information of latent fingerprints. The core network in the model is a conditional generative adversarial network that reconstructs the obscured ridge information of the latent samples. To overcome the limitation of the previous ridge-based reconstruction methods, our model predicts three extra maps in addition to the ridge map, namely the orientation, frequency, and segmentation maps. Generating the orientation and frequency maps ensure that the model is considering the orientation and frequency information of the input latent fingerprints. Generating a segmentation map prevents the model from filling large missing areas in the input latent samples; thus, it optimizes the amount of ridge information that can be reconstructed. In addition, to force the generator to preserve the ID information (type and location of minutiae), we developed an auxiliary deep model to extract the perceptual ID information (PIDI) of the generated sample and fuse it into the cGAN model to enhance the reconstruction process.

Chapter 2

Geometric Adversarial Attack on Deep Face Recognition

2.1 Introduction

Machine learning models especially DNNs have obtained SOTA performance in different domains [18, 47, 48].Despite the excellent performance, it has been shown [2, 32] that DNNs are vulnerable to a small perturbation in the input domain which can result in a drastic change of predictions in the output domain. These small perturbations, which are often imperceptible to humans, can transform natural examples into *adversarial examples* that are capable of manipulating high-level predictions of neural networks.

A crucial characteristic of adversarial examples is that they are visually similar to the original samples. This property significantly highlights the vulnerability of DNNs in critical applications where a carefully crafted adversarial example may remain benign to the human eye while targeting several machine learning models. For instance, autonomous vehicles may be misled by traffic signs constructed by an adversary to deceive machine learning methods, while the same sign may seem natural to human drivers [11].

Most of the attack methods developed in the previous works [2, 3, 49] are intensity-based attacks, as they directly manipulate the intensity of input images to fool the target model. Intensity-based attacks are computationally cheap and can prosper from a low-cost similarity constraint by adopting an ℓ_p-norm to force the generated examples to be similar to the

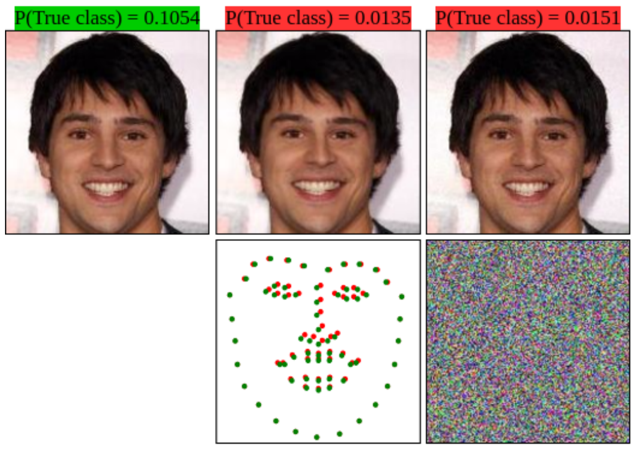


Figure 2.1: Comparison of the proposed attack to an intensity-based attack. First column: the ground truth image, which is correctly classified. Second column: the spatially transformed adversarial image wrongly classified and the corresponding adversarial landmark locations computed by our method. Third column: the adversarial image wrongly classified and the corresponding perturbation generated by the fast gradient sign method [2]. The proposed method leads to natural adversarial faces which are clean from additive noise.

benign samples. Since perturbations for neighborhood pixels are computed independently, adversarial examples generated using intensity-based attacks often have high-frequency components that can be used as a measure to detect and remove them [50]. On the other hand, the ℓ_{p}-norm is not a perfect measure for perceptual similarity since it is sensitive to spatial transformations [51]. For instance, a small rotation, translation, or scale variation in the input image, results in a drastic change of similarity. These limitations restrict intensity-based attacks from incorporating spatial perturbations. Recently, Xiao et al. [1] proposed a novel method of generating adversarial examples by spatially transforming natural images. Spatial transformations provide a convenient way of incorporating neighborhood information through interpolation.

From the defensive perspective, we divide FR systems into two different types, active and passive. In the active type, the model processes online face images from devices such as surveillance or access control cameras to identify the captured face. Therefore, the model has a limited amount of time to examine whether the input image is natural or not. In the passive FR, individuals submit a digital or hard copy photo to register their identity in a system for future identification. The attacker can submit an adversarial face image that prevents the system from recognizing the malicious ID in the future. In such a case, the defense algorithm has unlimited time to examine the gallery images. Hence, attacking passive FR systems is more challenging than attacking active systems. However, if the attack on the passive FR system is successful, the attacker may obtain a long-term immunity against the identification system.

This study explores the extent to which passive FR systems are vulnerable to spatially transformed adversarial examples. Inspired by [1], we propose a novel and fast method of generating adversarial faces by altering the landmark locations of the input images. The resulting adversarial faces completely lie on the manifold of natural images, which makes it extremely hard for defense methods to detect them even by a novelty detector [52]. The contributions of this paper are as follows:

- We have demonstrated that the prediction of a FR model has a linear trend around the actual value of the landmark locations of the input face image.
- We have introduced a fast method of generating adversarial face images, which is approximately 200 times faster than the previous geometry-based attacks which use L-BFGS optimization.
- We have developed a structure-constrained attack that manipulates face landmarks based on the semantic regions of the face.
- We have demonstrated that constraining the attack to preserve the natural structure of faces greatly increases the robustness of the method against the SOTA defense algorithms.

2.2 Related Work

Recent advances in technology have led to the generation of large datasets and powerful computational resources that made it possible to train deeper learning models. These models outperformed traditional methods in different areas ranging from signal processing to action

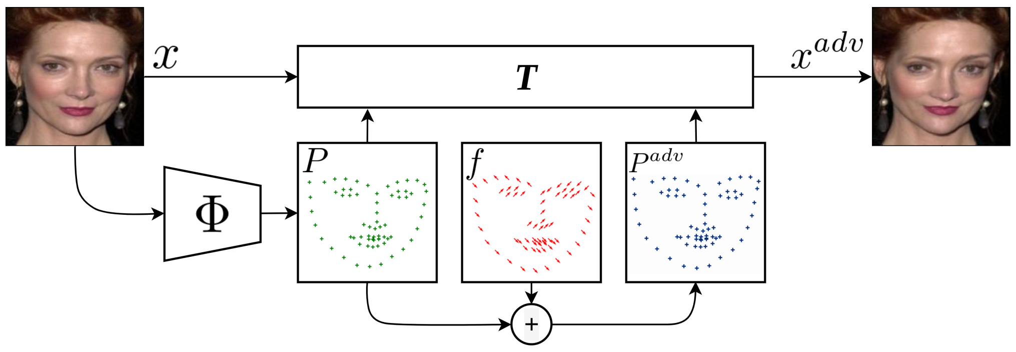


Figure 2.2: The proposed method optimizes a displacement field f to produce adversarial landmark locations $\mathrm{P^{adv}}$. The spatial transformation T transforms the input sample to the corresponding adversarial image $\mathrm{x^{adv}}$ such that $\boldsymbol{\Phi}(\mathrm{x^{adv}}) = \boldsymbol{\Phi}(\mathrm{x}) + \mathrm{f}$, and a SOTA FR model g miss-classifies the transformed image $\mathrm{x^{adv}}$.

recognition. Despite the spectacular performance, Szegedy et al. [32] showed that a small perturbation in the input domain can fool a trained classifier into making a wrong prediction confidently. In this section, we first review the literature on intensity-based and geometry-based attacks. Then we explore the background of adversarial examples for the FR systems.

2.2.1 Intensity-Based Attacks

Algorithms for generating adversarial examples can be categorized by the perturbation type. Most of the previously proposed methods are intensity-based attacks, as they directly try to manipulate the intensity of the input sample. Szegedy et al. [32] used a box-constrained L-BFGS [53] to generate some of the very first adversarial examples. Despite the high computational cost, their method was able to fool many networks trained on different inputs.

Goodfellow et al. [2] proposed a fast and efficient intensity-based attack called the fast gradient sign method (FGSM) and showed that the prediction of a deep leaning model has a linear trend around the saddle point of the input sample. Hence, they used the sign of the gradient of the classification loss with respect to the input sample as the perturbation to manipulate the intensity of the benign examples. This provides a fast and effective single-step attack. Although they select a small coefficient for the amplitude of the gradient sign to make the perturbation imperceptible, such a noisy pattern can facilitate the process of defending against it [50, 54]. Various extensions to intensity-based attacks have been developed to explore the vulnerability of machine learning models. To increase the effectiveness of the attack, Rozsa et al. [55] proposed to use the actual gradient value instead of the gradient sign used in FGSM [2]. Also, several iterative methods are developed to improve the robustness of single-step attacks against defenses, including the iterative version of FGSM [11, 56].

Papernot et al. [57] proposed the use of the Jacobian matrix of the prediction of classes concerning the input sample to generate *Jacobian-based Saliency Map Attack* (JSMA). JSMA reduces the number of pixels that are needed to be changed during the attack by calculating a saliency map of the most important pixels in the input space. Carlini and Wagner [58] modified the JSMA by changing the target layer used in the algorithm to compute the Jacobian matrix. They reported the adversarial success rate of 97% by modifying less than 5% of pixels in the input samples. However, saliency-based methods are computationally expensive due to the greedy search for finding the most significant areas in the input sample.

Almost all intensity-based attacks add high-frequency components to the input samples and use an ℓ_{p}-norm constraint to control the amount of distortion. However, the ℓ_{p}-norm is not a perfect similarity measure and does not guarantee that the adversarial samples lie on the same manifold as the natural samples. This increases the vulnerability of intensity-based attacks, especially in the passive applications where the agent has unlimited time to assess the legitimacy of the inputs.

2.2.2 Geometry-Based Attacks

Recently, Xiao et al. [1] proposed stAdv attack in which they generate adversarial examples by spatially transforming benign images. For this purpose, they define a flow field f for all pixel locations in the input image. The corresponding location of a pixel in the adversarial image can be computed by the displacement field. Since the displacement field can hold fractional values, they use a differentiable bilinear interpolation [59] to overcome the discontinuity problem. Furthermore, they added the sum of the total displacement of any two adjacent pixels to the main loss function to control the amount of distortion introduced by the displacement field. However, optimizing a flow field for all pixels in an image produces a highly non-convex cost function. They used the L-BFGS [53] with a linear backtrack search to find the optimal flow field f^*. Such a computationally expensive optimization is the critical limitation of this method.

2.2.3 Attacking Face Recognition

All of the previously proposed attack methods can be adopted for FR models, but the approach is highly dependent on the type of the FR model. For active FR, it has been shown that putting on enormous amounts of makeup [60] or wearing carefully crafted accessories [4] can conceal the identity of the attacker. However, wearing heavy makeup or overt accessories may draw attention and increase the chance of defense against the attack.

For passive FR, Goel et al. [61] and Goswami et al. [62] examined several intensity-based attacks and showed that they are extremely successful in fooling FR systems. However, the noisy structure of the perturbation makes these attacks vulnerable against conventional defense methods such as quantizing [54], smoothing [61] or training on adversarial examples [32].

2.2.4 Defense Methods

Since the introduction of adversarial examples, many approaches have been proposed to detect and mitigate these threats. Current defenses against adversarial attacks consist of two main approaches which modify either the model [2, 43, 63] or the input before feeding to the model [64, 65]. The most successful group of defenses to date are methods based on modifying the model, especially by using adversarial training [43].The adversarial training uses the adversarial examples during the training phase to make the model robust against the attack. Goodfellow et al. [2] proposed to utilize FGSM to generate adversarial examples and use them to train the model to provide robustness against adversarial examples. Later in Section 2.4.4, we use this method followed by ensemble adversarial training [63] and projected gradient descent [43] to examine the performance of our attacks under these SOTA defenses.

2.3 Approach

Here we first briefly describe the problem of generating adversarial examples. We then define a face transformation model based on the landmark locations in Section 2.3.2. We continue by presenting a landmark-based attack in Section 2.3.3 and developing a structural constraint in Section 2.3.4.

2.3.1 Problem Definition

For the process of generating adversarial faces, we assume that the victim FR model is a well-trained classifier $\mathrm{g} : \mathrm{x} \rightarrow \mathrm{y}$ over $\mathrm{N_c}$ different classes, that predicts a vector of classification scores $\mathrm{y} \in \mathbb{R}^{\mathrm{N_c}}$, given an input face image $\mathrm{x} \in [0,1]^{\mathrm{H}\times\mathrm{W}\times 3}$ with spatial size $\mathrm{H} \times \mathrm{W}$. We consider the white-box scenario where the attacker has full knowledge about the model and its prediction. The attacker tries to manipulate a benign face image x from class c in a way that the FR model miss-classifies the resulting adversarial face image $\mathrm{x}^{\mathrm{adv}}$.

2.3.2 Landmark-Based Face Transformation

Let $\boldsymbol{\Phi}$ be a landmark detector function that maps the face image x to a set of k 2D landmark locations $\mathrm{P} = \{\mathrm{p}_1, \ldots, \mathrm{p}_\mathrm{k}\}$, $\mathrm{p_i} = (\mathrm{u_i}, \mathrm{v_i})$. We assume $\mathrm{p}_\mathrm{i}^{\mathrm{adv}} = (\mathrm{u}_\mathrm{i}^{\mathrm{adv}}, \mathrm{v}_\mathrm{i}^{\mathrm{adv}})$ is the transformed version of $\mathrm{p_i}$, and defines the location of the i-th landmark in the corresponding adversarial face image $\mathrm{x}^{\mathrm{adv}}$. To manipulate the face image based on P, we define the per-landmark flow (displacement) field f to produce the location of the corresponding adversarial landmarks. For the i-th landmark $\mathrm{p}_\mathrm{i}^{\mathrm{adv}} = (\mathrm{u}_\mathrm{i}^{\mathrm{adv}}, \mathrm{v}_\mathrm{i}^{\mathrm{adv}})$, we optimize the spatial displacement vector $\mathrm{f_i} = (\Delta\mathrm{u_i}, \Delta\mathrm{v_i})$[*]. The adversarial landmark $\mathrm{p}_\mathrm{i}^{\mathrm{adv}}$ can be obtained from the original landmark $\mathrm{p_i}$ and the displacement vector $\mathrm{f_i}$ as:

$$\begin{aligned} \mathrm{p}_\mathrm{i}^{\mathrm{adv}} &= \mathrm{p_i} + \mathrm{f_i}, \\ (\mathrm{u}_\mathrm{i}^{\mathrm{adv}}, \mathrm{v}_\mathrm{i}^{\mathrm{adv}}) &= (\mathrm{u_i} + \Delta\mathrm{u_i}, \mathrm{v_i} + \Delta\mathrm{v_i}). \end{aligned} \tag{2.1}$$

[*]We assume that 2D coordinates are independent. So in the rest of the paper, all operations on coordinates are element-wise.

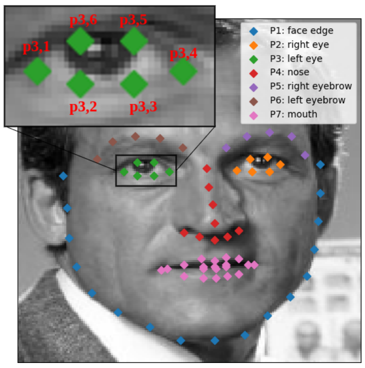


Figure 2.3: Grouping face landmarks based on semantic regions of the face.

Contrary to [1], which estimates the displacement field f for all pixel locations in the input image, the displacement field f in the proposed method is only defined for k landmarks. In a real-world application, especially FR problems, k is notably small compared to the number of pixels in the input image. As a result, it is possible to use conventional spatial transformations to transform the input image. Consequently, limiting the number of control points reduces the distortion introduced by the spatial transformation. The resulting adversarial face image is the transformed version of the benign face image using the transformation T as follows:

$$\mathrm{x}^{\mathrm{adv}} = \mathrm{T}(\mathrm{P}, \mathrm{P}^{\mathrm{adv}}, \mathrm{x}), \tag{2.2}$$

where T is the spatial transformation that maps the source control points P to the target control points $\mathrm{P}^{\mathrm{adv}}$. Note that $\mathrm{x}^{\mathrm{adv}}$ is differentiable with respect to the landmark locations and the input image.

2.3.3 Fast Landmark Manipulation

It has been shown [66] that landmark locations in the face image provide highly discriminative information for FR tasks. Indeed, we experimentally show this in Section 2.4.2 that even learning based FR systems discriminate face identities based on extracting the relative geometric features. More specifically, the predictions of FR systems are highly linear around the original landmark locations of the face image. This property allows the direct employment of the gradient of the prediction in a FR model to geometrically manipulate benign faces.

We use the gradients of the prediction with respect to the location of landmarks to update the displacement field f. For this purpose, we first define a standard for the correct prediction, and then we use it to compute the formulation of the attack. As a measure of correct classification, we select the same *softmax cost* used in [4, 67]. Given an input x, a one-hot label vector $\mathrm{y_c}$ corresponding to class c and a vector of classification score g(x) from the victim classification model, we define the *softmaxcost* as:

$$\mathrm{J}\Big(\mathrm{g(x), c}\Big) = -\log\left(\frac{e^{\mathrm{y_c^T g(x)}}}{\sum_{\mathrm{n=1}}^{\mathrm{N_c}} e^{\mathrm{y_n^T g(x)}}}\right), \tag{2.3}$$

where $\mathrm{N_c}$ is the number of classes. Besides, we define a boundary for the amount of displacement to prevent the model from generating distorted face images. Inspired by [1], we develop $\mathrm{L_{flow}}$ to constrain the displacement field f as follows:

$$\mathrm{L_{flow}(f)} = \frac{1}{\mathrm{k}}\sum_{\mathrm{i=1}}^{\mathrm{k}}(\Delta \mathrm{u_i}^2 + \Delta \mathrm{v_i}^2). \tag{2.4}$$

Having the measure for the correct classification, and the term for bounding the displacement field, we define the total loss for generating adversarial faces as:

$$\mathrm{L_t(P, P^{adv}, x, c_x)} = \mathrm{J}\Big(\mathrm{g}\Big(\mathrm{T(P, P^{adv}, x)}\Big), \mathrm{c_x}\Big) - \lambda_{\mathrm{flow}}\mathrm{L_{flow}(P^{adv} - P)}, \tag{2.5}$$

where λ_{flow} is a positive coefficient used to control the magnitude of the displacement. The attacker can generate geometric adversarial perturbations by finding the f^* as:

$$\mathrm{f}^* = \arg\max_{\mathrm{f}} \mathrm{L_t(P, P^{adv}, x, c_x)}. \tag{2.6}$$

As we show in Section 2.4.2, the prediction of FR models is highly linear around the ground truth location of the face landmarks; therefore, we use the direction of the gradient of the prediction (same as FGSM [2]) to find the landmark displacement field f in an iterative manner. The t-th optimization step for finding f using FGSM is:

$$\mathrm{f^{(t+1)}} = \mathrm{f^{(t)}} + \varepsilon\,\mathrm{sign}(\nabla_{\mathrm{P^{adv(t)}}}\mathrm{L_t(P, P^{adv(t)}, x, c_x)}), \tag{2.7}$$

where $\mathrm{P}^{\mathrm{adv}(t)} = \mathrm{P} + \mathrm{f}^{(t)}$. We refer to this as the *fast landmark manipulation method* (FLM) for generating adversarial faces. Figure 2.2 shows an overview of the method.

2.3.4 Semantic Grouping of Landmarks

In the previous section, we developed a model to generate face images based on manipulating the landmark information. Although this method is fast and computationally cheap, it has a limitation that should be addressed. In Equation 2.7, we use the gradients of the classification loss with respect to the landmark locations to update the displacement field for generating the adversarial face images. These gradients can have any direction in the 2D coordinate space. As a result, multiple updates of the displacement field f can severely distort the generated adversarial images. To prevent this issue, we adopt the total ℓ_2–norm of the displacement field f as an additional loss. However, our model computes the displacement field f for a significantly small number of locations in the input image, so limiting the size of f can reduce the effectiveness of the attack.

To overcome this limitation, we propose to semantically group landmarks and manipulate the group properties instead of perturbing each landmark. Consequently, the total structure of the face will be preserved. This consideration allows us to increase the total amount of displacement and, as a result, extremely increases the effectiveness of the attack. We break down the set of landmarks P into m semantic groups $\mathrm{P}_i, i \in \{1, \dots, m\}$, and $\mathrm{p}_{i,j}$ denotes the j-th landmark in the i-th group which has n_i landmarks. These groups are formed based on their semantic regions in the face, such as left eye, right eye, mouth, etc.. Figure 2.3 shows a sample grouping of face landmarks used in this study. We define a flow field vector for each of the groups by means of a translation and a scale variable that will apply to all elements in the group. For the face regions, a rotation is not of interest because it is not natural to have a face with a rotated mouth or nose.

Let P_i be the i-th landmark group e.g. all landmarks of the nose. To scale these landmarks, we define the scaling tuple $\alpha_i = (\alpha_{u_i}, \alpha_{v_i})$ where α_{u_i} and α_{v_i} are the horizontal and vertical scaling parameters respectively. To translate the landmarks, we define the translation tuple $\beta_i = (\beta_{u_i}, \beta_{v_i})$ where β_{u_i} and β_{v_i} are the translation parameters for the horizontal and vertical axes respectively. The location of the corresponding landmarks in the adversarial image can be computed as:

$$\mathrm{P}_{\mathrm{i}}^{\mathrm{adv}} = \alpha_{\mathrm{i}}(\mathrm{P}_{\mathrm{i}} - \overline{\mathrm{p}_{\mathrm{i}}}) + \beta_{\mathrm{i}}, \tag{2.8}$$

where $\overline{\mathrm{p}_{\mathrm{i}}}=\frac{1}{\mathrm{n_i}}\sum_{\mathrm{j}=1}^{\mathrm{n_i}} \mathrm{p}_{\mathrm{i,j}}$ is the average location of all landmarks in the group $\mathrm{P_i}$. We subtract the average of the group from each landmark location in the group before scaling to force each part of the face to be scaled regarding its center.

We choose α_{i} and β_{i} such that they minimize the square error of $\mathrm{P}_{\mathrm{i}}^{\mathrm{adv}}$ between Equation 2.1 and Equation 2.8 as:

$$\underset{\alpha_{\mathrm{i}},\beta_{\mathrm{i}}}{\arg\min}\ \frac{1}{\mathrm{n_i}}\sum_{\mathrm{j}=1}^{\mathrm{n_i}}\Big(\alpha_{\mathrm{i}}(\mathrm{p}_{\mathrm{i,j}} - \overline{\mathrm{p}_{\mathrm{i}}}) + \beta_{\mathrm{i}} - \mathrm{p}_{\mathrm{i,j}} - \mathrm{f}_{\mathrm{i,j}}\Big)^2. \tag{2.9}$$

Solving Equation 2.9 results in the closed-form solutions for the α_{i} and β_{i} as:

$$\alpha_{\mathrm{i}} = \frac{\sum_{\mathrm{j}=1}^{\mathrm{n_i}}(\mathrm{p}_{\mathrm{i,j}} - \overline{\mathrm{p}_{\mathrm{i}}})(\mathrm{p}_{\mathrm{i,j}} + \mathrm{f}_{\mathrm{i,j}})}{\sum_{\mathrm{j}=1}^{\mathrm{n_i}}(\mathrm{p}_{\mathrm{i,j}} - \overline{\mathrm{p}_{\mathrm{i}}})^2}, \tag{2.10}$$

$$\beta_{\mathrm{i}} = \overline{\mathrm{p}_{\mathrm{i}}} + \frac{1}{\mathrm{n_i}}\sum_{\mathrm{j}=1}^{\mathrm{n_i}} \mathrm{f}_{\mathrm{i,j}}. \tag{2.11}$$

We modeled the effect of the displacement field $\mathrm{f}_{\mathrm{i,j}}$ for each group of landmarks as a scaling and a translation function. While Equation 2.7 optimizes f, we use Equations 2.10 and 2.11 to calculate the corresponding set of scale tuples $\{\alpha_1, \ldots, \alpha_7\}$ and translation tuples $\{\beta_1, \ldots, \beta_7\}$. We refer to this as the *grouped fast landmark manipulation method* (GFLM) for generating adversarial faces.

2.4 Experiments

We first describe the implementation details in Section 2.4.1. Then we investigate how landmark information influences the prediction of a face classifier in Section 2.4.2. We evaluate the performance of the proposed attacks in the white-box scenario in Section 2.4.3 and conclude the experiments by measuring and comparing the performance of our attacks under several defense methods in Section 2.4.4.

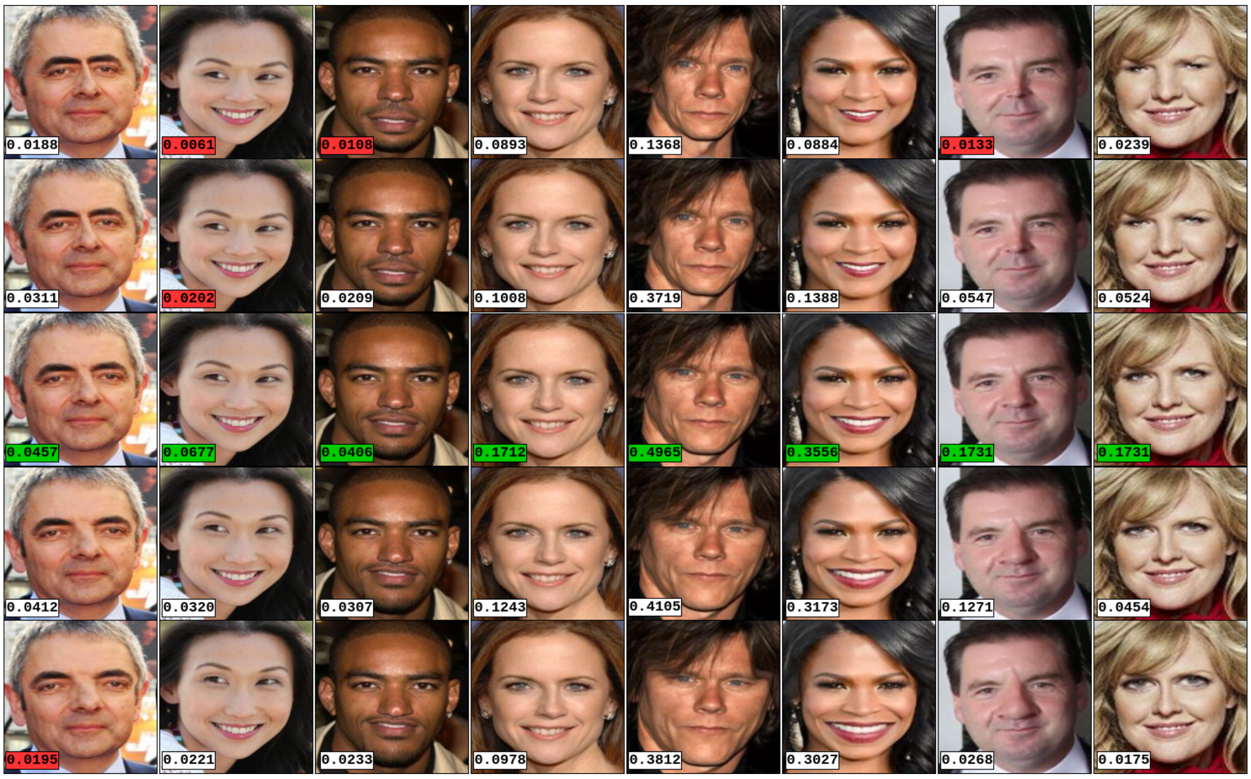


Figure 2.4: Examples of linearly interpolating face properties. Each column from Left to right shows examples of interpolating one of the eight geometric variables of the face structure described in Section 2.4.2. The probability of the true class is depicted on the bottom left corner of samples. The green color specifies the face image with the maximum probability of belonging to the true class. The red color shows the incorrectly classified face images.

2.4.1 Implementation Details

To evaluate the performance of the proposed method in the white-box scenario, we use the FR model developed by Schroff et al. [68] that obtained the SOTA results on the Labeled Faces in the Wild (LFW) [7] challenge as the victim model. We train two instances of the model* on two datasets of face images. The first instance is trained to recognize 9,101 celebrities from the VGGFace2 dataset [69] with more than 3.3M training images and the average of 360 images per subject. The second instance is trained on the CASIA-WebFace [70] dataset which consists of more than 494,000 face images and 10,575 unique

*https://github.com/davidsandberg/facenet

IDs. For extracting the landmark information of the input face images, we use the Dlib [71] landmark detector which predicts the 2D coordinates for 68 landmarks. We divide landmarks based on five facial regions as: 1) P_1: *jaw*, 2) P_2: *right eye and eyebrow*, 3) P_3: *left eye and eyebrow*, 4) P_4: *nose*, and 5) P_5: *mouth*. The number of landmarks in each group is as: $\{n_1{=}17, n_2{=}11, n_3{=}11, n_4{=}9, n_5{=}20\}$. Figure 2.3 demonstrates a similar grouping of landmarks.

We opt to use the thin-plate spline (TPS) [72] to cover a broad range of spatial transformations that are capable of locally manipulating face images. TPS has $2(k + 3)$ parameters for mapping k source landmarks P to their corresponding P^{adv}. We first scale coordinates to lie inside the range $[-1, 1]^2$ where $(-1, -1)$ is the top left corner and $(1, 1)$ is the bottom right corner of the image. We assume all coordinates are continuous values since TPS has no restriction on the continuity of the coordinates because of the differentiable bilinear interpolation [59].

We set the value of λ_{flow} for the fast landmark manipulation method (FLM) attack to 100. For the grouped fast landmark manipulation method (GFLM) we do not set any limit for the amount of displacement since the structural condition developed in Section 2.3.4 is enough to preserve the similarity of the generated adversarial examples. Therefore, we set λ_{flow} for the GFLM attack to zero. To further condition the model to generate realistic faces, we perform an extra modification for the symmetric parts, such as eyes. We set an equal scale and an equal vertical position for these parts. Other conditions can be applied by slightly changing Equation 2.9. For example, instead of manipulating the horizontal location of the eyes independently, one can change the horizontal distance between them to preserve the natural symmetry.

2.4.2 Interpolated Perturbation

The geometry of the face is unique and provides highly discriminative information for FR. In this section, we perform an experiment to evaluate how spatially manipulating the face regions affects the performance of a FR system. We extract landmarks for all faces in the CASIA-WebFace [70] dataset and define eight variables based on the geometric properties

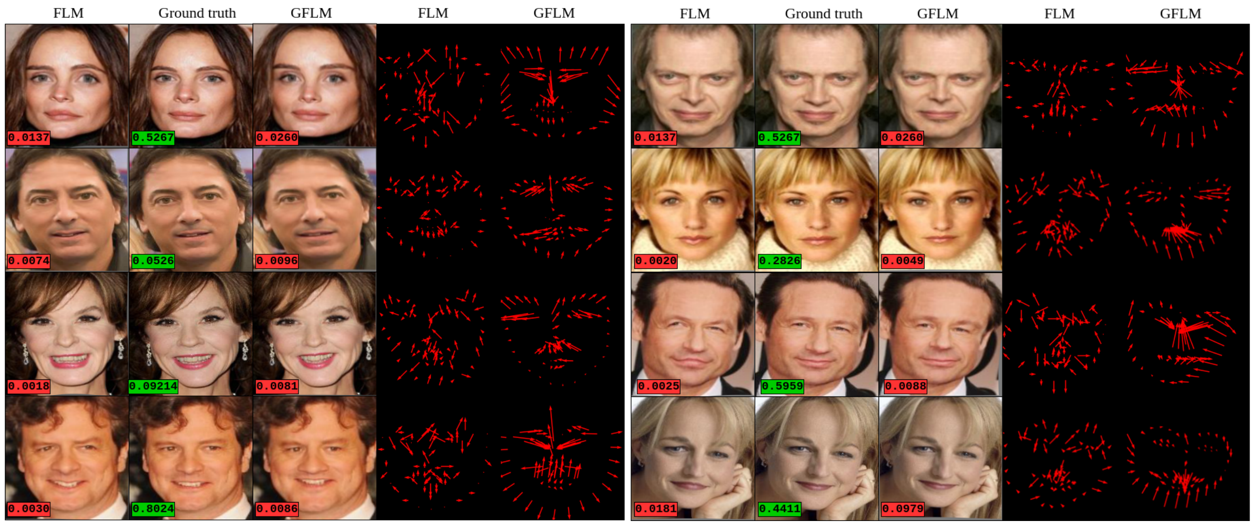


Figure 2.5: Examples of the adversarial faces generated using FLM and GFLM. For each subject, five images are shown including the original face image (middle face), the result of GFLM (right face), the result of FLM (right image), displacement field f for GFLM (left field) and displacement field f for FLM (right field). Tags on the bottom left of images show the probability of the true class. Green and red tags denote the correct and incorrect classified samples respectively.

of the face regions. The first four variables are the translation-based variables which are: 1) horizontal distance between the eyes and eyebrows, 2) vertical location of the eyes and eyebrows, 3) horizontal location of the nose, 4) horizontal location of the mouth. The second set of variables are the scale-related variables and are as follows: 5) scale of the jaw, 6) scale of the mouth, 7) scale of the nose, and 8) scale of the eyes. We interpolated each of these variables independently to measure the influence of each on the performance of the FR model. Figure 2.4 shows several examples of the interpolation.

We calculate the prediction of the true class for faces which are correctly classified and their manipulated versions. The predictions are averaged over all the ID's to investigate how manipulating face parts affects the predicted probability of the class. Figure 2.6 shows the final averaged values for the predictions. As it is shown, the global maximum of the model's prediction for a sample face is around the ground truth value of the positions and the scales. These results confirms that the geometry of the face contains highly discriminative

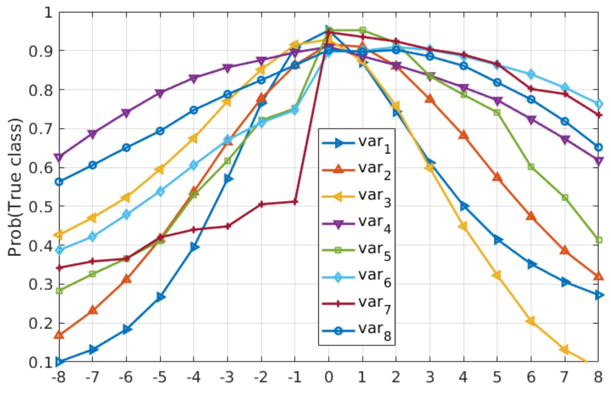


Figure 2.6: Normalized probability of the true classes based on interpolating the eight variables of face geometry defined in Section 2.4.2.

information for FR. Indeed, the prediction of a FR model has a linear characteristic around the actual size and location of face regions and enables us to directly use the gradient of the prediction to manipulate landmark locations.

#	Face	**FLM**				**GFLM**				**stAdv [1]**		
	Region	$\bar{n}$	SR(%)	pT	T(s)	$\bar{n}$	SR(%)	pT	T(s)	SR(%)	pT	T(s)
1	Eyebrows	9.6	61.92	0.0156	0.473	21.1	79.45	0.0159	1.019			
2	Eyes	11.5	55.59	0.0169	0.511	18.6	83.06	0.0156	0.781			
3	Nose	8.8	82.50	0.0139	0.495	10.7	89.37	0.0149	0.575			
4	Mouth	10.4	49.27	0.0152	0.455	20.2	77.13	0.0165	0.868			
5	Jaw	8.7	12.54	0.0180	0.420	37.6	49.26	0.0161	1.804			
6	All	**2.8**	99.86	**0.0072**	**0.126**	5.2	**99.96**	0.0120	0.254	99.18	0.0166	27.177

Table 2.1: Comparing results of the proposed attacks to stAdv [1] and exploring the influence of different regions of the face on our attacks. In each experiment, the average number of iterations ($\bar{n}$), the success rate of the attack(SR), the average final probability of the true class (pT), and the average time of the attack are shown.

Defense	FGSM [2]	stAdv [1]	FLM	GFLM
Adv. [2]	19.12	36.96	54.79	**62.03**
Ens. [63]	16.27	33.80	53.59	**61.84**
PGD [43]	18.95	39.15	55.65	**67.43**

Table 2.2: Comparing the success rate of the proposed FLM and GFLM attacks to FGSM [2] and stAdv [1] attacks under the SOTA adversarial training defenses.

2.4.3 White-Box Attack

We evaluate the performance of both proposed methods of FLM and GFLM for the white-box attack scenario on the CASIA-WebFace [70] dataset. We define six experiments to investigate the importance of each region of the face in the FLM and GFLM attack methods. In the first five experiments, we evaluate the performance of the attacks on each of the five main regions of the face including 1) eyebrows, 2) eyes, 3) nose, 4) mouth and 5) jaw. In the last experiment, we evaluated the performance of attacks using all five regions of the face. Also, in Experiment 6, we compare the performance and speed of the proposed methods to the method developed by [1] in which the displacement field f is defined for all pixels in the input image. All the experiments are conducted on a PC with 3.3 GHz CPU and NVIDIA TITAN X GPU. Table 2.1 shows the results for all the six experiments.

From the results, we observe that both the FLM and GFLM are generating powerful adversarial face images that fool the classifier for more than 99.86% of the samples. An important point is the computation time of these algorithms. The average time of generating adversarial faces for the FLM and GFLM is 125 and 254 milliseconds respectively, which is significantly shorter than the computation time of stAdv [1], which is 27.177 seconds on average. Indeed, the FLM is 215 and GFLM is 106 times faster than stAdv [1] method. Furthermore, we described in Section 2.3.4 that the FLM can generate faces with spatial distortions, and grouping the landmarks in the GFLM overcomes this problem. Figure 2.5 demonstrates several examples of the adversarial faces generated by the FLM and GFLM.

2.4.4 Performance Under Attacks

To evaluate the performance of the proposed methods under attack, we repeat the sixth experiment in the previous section. For this purpose, we use three SOTA defenses of FGSM adversarial training [2], PGD adversarial training [43], and ensemble adversarial training [63]. We compare the performance of our attacks to FGSM [2] and stAdv [1]. Results are shown in Table 2.2. The FLM and GFLM attacks are extremely robust against adversarial training compared to FGSM [2] and stAdv [1] because they are targeting the most important locations in the benign samples using geometric perturbations. These locations contain the most critical discriminative information that a FR model needs to identify an individual. Defenses based on adversarial training use the intensity-based attacks to generate samples for training the model. However, the generated samples do not lie on the manifold of natural images due to the slight change of intensity of all pixels in the input image. Furthermore, the GFLM is more robust against defenses than the FLM since samples generated by the GFLM are conditioned to have the similar structure as a natural face.

2.5 Conclusion

In this paper, we introduced a novel method for generating adversarial face images by manipulating landmark locations of the natural images. Landmark locations contain highly discriminative information for face identification. Therefore, manipulating landmark locations is a strong way to change the prediction of a FR system. We experimentally showed that the prediction of a FR model has a linear trend around the parameters of the model and the landmark locations of the input image. This finding indicates that one can directly manipulate landmark locations using the gradient of the prediction with respect to the input image.

Based on this idea, we introduced a fast method of manipulating landmark locations through spatial transformation, which is approximately 200 times faster than the previous geometric attacks, with the success rate of 99.86%. In addition, we developed a second attack constrained on the semantic structure of the face. The second attack is extremely powerful in generating natural-looking samples that are hard to detect even for the SOTA defense methods.

Chapter 3

Smooth Adversarial Perturbations

3.1 Introduction

Despite revolutionary achievements of DNNs in many computer vision tasks [18,73–75], carefully manipulated input samples, known as *adversarial examples*, can fool learning models to confidently make wrong predictions [32]. Adversarial examples are potential threats to almost all applications of machine learning [34–36], but the case is more severe in the context of computer vision, particularly, due to the complexity of tasks [39], huge cardinality of input spaces [76], and sensitivity of applications [77–80]. Analyzing DNNs as differentiable transfer functions have led to substantial studies exploring embedding spaces and their characteristics in regard to training paradigms. However, the adversarial behavior has highlighted the importance of studying the topology of decision boundaries and their properties in high dimensional data spaces [39,40].

Considering a *white-box* scenario where the network architecture and all its parameters are known, several approaches (attacks) have been proposed to explore the robustness of decision boundaries in the presence of ℓ_{p}-bounded [10,11,32] and ℓ_{p}-minimal [3,57,58,81,82] adversarial perturbations. However, the vulnerability of DNNs to adversarial perturbations with specific statistical properties or frequency-domain characteristics, which lie beyond the conventional ℓ_{p}-norm constraints, has remained less explored.

In this study, we seek to explore the landscape of robustness of DNNs to adversarial perturbations with modified frequency-domain characteristics. Specifically, we focus on smooth

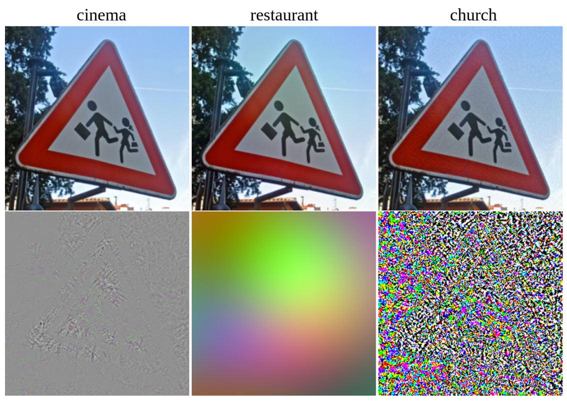


Figure 3.1: Comparing smooth adversarial perturbations with conventional adversarial perturbations. Each column from left to right shows the adversarial image and the corresponding adversarial perturbations computed by DeepFool [3], SmoothFool ($\boldsymbol{\sigma}_{\mathbf{g}} = 75$) and IGSM [11], respectively, on ResNet-101 [12]. The predicted label for each image is depicted above the column. Perturbations are magnified for a better visibility.

adversarial perturbations due to several advantages they offer compared to the conventional adversarial perturbations. First, they are more physically realizable than non-smooth adversarial perturbations since printing devices are critically less accurate in capturing high frequency structures due to the sampling noise [4]. Also, severe differences between adjacent pixels in the printed adversarial examples are unlikely to be accurately captured by cameras due to their low-pass spatial frequency response [83]. Second, the high-frequency structure of conventional adversarial perturbations has provoked an intensive adoption of explicit [50,84,85] and implicit [76,86,87] denoising methods to mitigate the adversarial effect. However, we demonstrate that a slight modification of local statistics of adversarial perturbations causes a vital failure of SOTA defenses. Third, smoothness significantly enhances the transferability of adversarial perturbations across classifiers and data points by improving the invariance of perturbations to translation [88]. This improves the performance of the attack in the *black-box* scenario where the parameters of the target model are not known to the adversary. Forth, smoothness enhances plausible deniability and allows the attacker to disguise adversarial perturbations as natural phenomena such as shadows. In this way, the magnitude of adversarial perturbations can be increased notably since imperceptibility is less important.

We formulate the problem of constructing smooth adversarial perturbations according to a general definition of smoothness and exploit the geometry of decision boundaries to find computationally efficient solutions. Our main contributions are the followings:

- We propose SmoothFool, a geometry inspired framework for computing smooth adversarial perturbations which exploits the topology of decision boundaries to find efficient adversarial perturbations.
- We analyze various properties of smooth adversarial perturbations and validate their effectiveness for both the white-box and black-box attack scenarios.
- We show the susceptibility of two major group of defenses against smooth adversarial perturbations by breaking several SOTA defenses.
- We integrate SmoothFool with previous studies on universal adversarial perturbations and demonstrate the existence of smooth universal adversarial perturbations that generalize well across data samples and network architectures.

3.2 Related Work

3.2.1 Adversarial Attack

Despite the highly non-linear nature of DNNs, they have been observed to exhibit linear characteristics around the actual parameters of the model and the input samples [10, 89, 90]. In particular, Goodfellow et al. [10] showed that the prediction of DNNs can be changed drastically by translating the input sample toward the gradient of the classification loss. Hence, they proposed the fast gradient sign method (FGSM) as a single step attack incorporating solely the sign of gradients to craft adversarial perturbations. Kurakin et al. [11] improved the performance of FGSM by adopting an iterative procedure called IGSM. Moosavi et al. [3] proposed DeepFool to find approximately ℓ_{p}-minimal adversarial perturbations by iteratively translating input samples toward the linearized approximation of the closest decision boundary. Our methodology builds on DeepFool to find minimal smooth adversarial perturbations.

Some prior studies have considered smoothness in adversarial attacks. Sharif et al. [4] added a total variation (TV) loss to the main objective of the attack to enhance the physical realizability of the resulting perturbations and showed that smoothness of adversarial perturbations improves their effectiveness for the real-world applications. Fong et al. [91] demonstrated that smoothing important regions in the input example can deteriorate the confidence of prediction. They utilized this observation to interpret the decisions of DNNs. Hosseini et al. [6] proposed constructing semantic adversarial examples by randomly shifting Hue and Saturation components of benign samples in the HSV color space. Dong et al. [88] demonstrated that robustness of DNNs to slight translations can be exploited to improve the trasferability of adversarial examples. Interestingly, the final perturbations crafted using their approach exhibited low-pass frequency response. However, their methodology is applicable for a limited level of smoothness since the prediction of DNNs is invariant for solely small translations of the input sample.

Fundamentally, our work differs from previous approaches since we seek to find approximately ℓ_2-minimal adversarial perturbations capable of offering arbitrary levels of smoothness. Also, our main goal is to formulate and compute smooth adversarial perturbations, not to find smooth adversarial examples, since the latter can critically destroy the structure of images.

3.2.2 Defense Methods

Since the first observation of adversarial perturbations, their noisy structure has been harnessed to find defense strategies. Several studies have incorporated explicit denoising techniques to mitigate the adversarial effect. Liao et al. [50] showed that the distribution of high-level representations in DNNs provides an effective guidance to denoise adversarial examples and proposed the high-level representation guided denoiser (HGD). Training DNNs using adversarial examples, known as *adversarial training* [10, 76, 86], has been shown to provide a relative adversarial robustness. Adversarial training can be considered as an implicit denoising technique which reduces the sensitivity of predictions to slight changes in the input domain. Manifold learning is another implicit denoising defense. A well-known example for this type of defense is *MagNet* [92] which deploys autoencoders for mapping input examples onto the manifold of natural examples. Later, we utilize these defenses to evaluate the effectiveness of smooth adversarial perturbations.

3.3 Smooth Adversarial Perturbations

3.3.1 Problem Definition

Let $\mathrm{f} : \mathbb{R}^{\mathrm{n}} \rightarrow \mathbb{R}^{\mathrm{m}}$ be a classifier mapping input sample $\mathbf{x} \in [0,1]^{\mathrm{n}}$ to m classification scores $\mathrm{f_j}(\mathbf{x})$, associated with each class $\mathrm{j} \in \{0, \ldots, \mathrm{m}-1\}$. The class predicted by the network can be computed as:

$$\mathrm{c}(\mathbf{x}) = \arg\max_{\mathrm{j}} \mathrm{f_j}(\mathbf{x}). \tag{3.1}$$

The problem of constructing smooth adversarial perturbations can be formulated as the following optimization problem:

$$\begin{aligned} \arg\min_{\mathbf{r}} \; & ||\mathbf{r}||_2 + \lambda\Omega(\mathbf{r}) \text{ subject to:} \\ & 1. \;\; \mathrm{c}(\mathbf{x}+\mathbf{r}) \neq \mathrm{c}(\mathbf{x}), \\ & 2. \;\; \mathbf{x}+\mathbf{r} \in [0,1]^{\mathrm{n}}, \end{aligned} \tag{3.2}$$

where $\mathbf{r} \in \mathbb{R}^{\mathrm{n}}$ is the AP, $\Omega(.)$ is a measure of roughness, and λ is a Lagrangian coefficient controlling the trade-off between roughness and magnitude of the perturbation. Generally, the roughness of perturbations can be defined based on their local variations. Such variations have an explicit interpretation in the frequency domain where the power of each frequency component captures the specific range of variations. Considering this perspective, we use a frequency response function H to formulate the definition of roughness since it can denote how much each frequency component contributes to the intended roughness. For clarity, we substitute H with $\mathrm{H_{hp}}$ to highlight the high-frequency nature of roughness and denote $\mathrm{H_{lp}} = 1-\mathrm{H_{hp}}$ as the complementary low-pass filter which defines the equivalent smoothness. We use the total energy of the high-frequency components of $\mathbf{r}$ as a general measure of roughness, and define Ω as:

$$\Omega(\mathbf{r}, \mathrm{H_{lp}}) := \int_{-\infty}^{+\infty} \mathrm{R}(\omega)^2(1-\mathrm{H_{lp}}(\omega))^2 \mathrm{d}\omega, \tag{3.3}$$

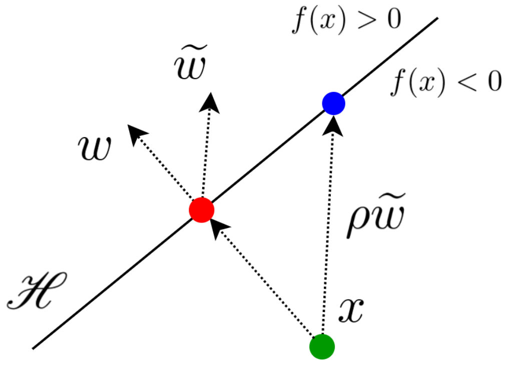


Figure 3.2: Finding smooth AP for a linear binary classifier. Red and blue dots show the ℓ_2 projection and smooth projection of $\mathbf{x}$ onto the decision boundary, respectively. For an easier demonstration, $\mathbf{x}$ is assumed to belong to class -1.

where R is the Fourier transformation of perturbation $\mathbf{r}$, and $\mathrm{H_{lp}}$ is the frequency response of a given low-pass filter defining the range of acceptable smoothness, and is a free parameter of the definition.

The perturbation $\mathbf{r}$ in our problem is represented as a set of spatially discrete adversarial perturbations for each pixel location $\mathrm{u} \in \{0, \dots, \mathrm{n}-1\}^*$, and $\mathbf{\Omega}$ can be conveniently computed in the spatial domain as:

$$\mathbf{\Omega}(\mathbf{r}; \mathbf{h}) = ||\mathbf{r} - \mathbf{r} * \mathbf{h}||_2^2, \tag{3.4}$$

where $*$ denotes convolution, and $\mathbf{h}$ is the discrete approximation of $\mathrm{H_{lp}}$ in the spatial domain. In the rest of the paper, our work builds on this definition of roughness (and, equivalently, smoothness) and aims to find adversarial perturbations which are relatively smooth based on any predefined $\mathbf{h}$ compared to perturbations crafted by other contemporary attacks. Due to the non-convex nature of the problem, we exploit the geometric properties of the decision boundary of DNNs to find a relaxed solution for the optimization problem given in Equation 3.2.

*Here we assume the input image $\mathbf{x}$ is a 1D signal, and later in the experiments we adopt all formulations for 2D images.

3.3.2 Linearized solution

Based on previous findings [3, 89, 93], the decision boundary of a differentiable classifier, f, around $\mathbf{x}$ can be well approximated by a hyperplane passing through the minimal ℓ_2 adversarial example $\mathbf{x}_p$ corresponding to $\mathbf{x}$, and the normal vector $\mathbf{w}$ orthogonal to the decision boundary at $\mathbf{x}_p$ as $\mathscr{H} \triangleq \{\mathbf{x} : \mathbf{w}^\top(\mathbf{x} - \mathbf{x}_p) = 0\}$. We assume $\mathbf{x}_p$ and, consequently, $\mathbf{w}$ associated with each $\mathbf{x}$ is available, and later we utilize an appropriate contemporary attack to compute $\mathbf{x}_p$ and $\mathbf{w}$. Having $\mathbf{x}_p$ provides two benefits. First, it allows us to linearize the closest decision boundary around $\mathbf{x}$. Second, we can reduce the problem to a binary classification problem, where the goal of the attack would be to compute the smooth perturbation $\mathbf{r}$ which yields $c(\mathbf{x}+\mathbf{r}) = c(\mathbf{x}_p)$. Consequently, we rewrite the optimization problem given in Equation 3.2 as:

$$
\begin{aligned}
&\arg\min_{\mathbf{r}} \; ||\mathbf{r}||_2 + \lambda\Omega(\mathbf{r}; \mathbf{h}) \text{ subject to:} \\
&\qquad 1.\;\; \mathbf{w}^\top(\mathbf{x} + \mathbf{r}) - \mathbf{w}^\top \mathbf{x}_p = 0, \\
&\qquad 2.\;\; \mathbf{x} + \mathbf{r} \in [0, 1]^n.
\end{aligned}
\tag{3.5}
$$

In this setup, an efficient solution can be obtained from a smooth projection of $\mathbf{x}$ onto the estimated hyperplane $\mathscr{H}$. Such a projection can be computed by translating $\mathbf{x}$ using the adversarial perturbation $\mathbf{r} = \rho\widetilde{\mathbf{w}}$, where $\widetilde{\mathbf{w}}$ is a smooth approximation of $\mathbf{w}$, and ρ scales $\widetilde{\mathbf{w}}$ to map $\mathbf{x} + \mathbf{r}$ on $\mathscr{H}$ as:

$$
\rho = \frac{\mathbf{w}^\top(\mathbf{x}_p - \mathbf{x})}{\mathbf{w}^\top\widetilde{\mathbf{w}}}. \tag{3.6}
$$

Figure 3.2 provides a simple visualization of this projection. It worth mentioning that for the linear binary classifier choice of f, the optimal smooth perturbation has the closed-form solution: $\mathbf{r} = -\frac{f(\mathbf{x})}{\mathbf{w}^\top\widetilde{\mathbf{w}}}\widetilde{\mathbf{w}}$. Generally, the estimation $\widetilde{\mathbf{w}}$ must hold two conditions to provide a valid solution for the linearized problem. First, $\widetilde{\mathbf{w}}$ should not be orthogonal to $\mathbf{w}$. Second, the estimation should remove high-frequency components of $\mathbf{w}$ in order to keep $\Omega(\rho\widetilde{\mathbf{w}}; \mathbf{h})$ low. Without loss of generality, we consider a low-pass filter $\mathbf{g}$ to estimate $\widetilde{\mathbf{w}}$ by convolution as: $\widetilde{\mathbf{w}} = \mathbf{g} * \mathbf{w}$, since it is easy to compute, and the only condition on $\mathbf{g}$ is that its cut-off frequency should be less than the cut-off frequency of $\mathbf{h}$.

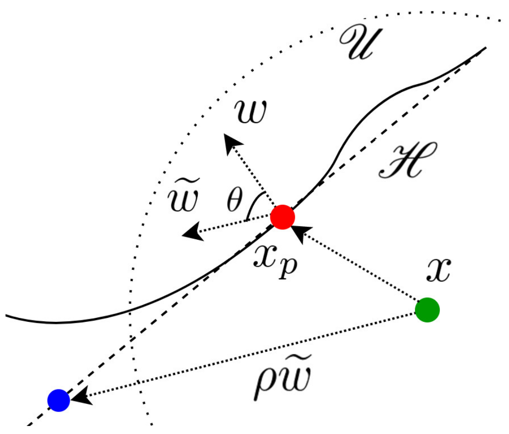


Figure 3.3: A demonstration of the topology of the decision boundary in the vicinity of data point $\mathbf{x}$. $\mathscr{U}$ illustrates the region where the decision boundary can be assumed to be approximately flat. Smooth projection of $\mathbf{x}$ onto the estimated hyperplane $\mathscr{H}$ often results in a solution out of $\mathscr{U}$.

The final smooth perturbation that can project $\mathbf{x}$ on $\mathscr{H}$ can be computed as:

$$\mathbf{r} = \frac{\mathbf{w}^\top(\mathbf{x}_\mathrm{p} - \mathbf{x})}{\mathbf{w}^\top(\mathbf{g} * \mathbf{w})}(\mathbf{g} * \mathbf{w}). \tag{3.7}$$

In this formulation, the cut-off frequency of $\mathbf{g}$ is associated with λ in the optimization problem given in Equation 3.5 since it controls the smoothness of perturbation $\mathbf{r}$. ℓ_2-DeepFool [3] constructs adversarial examples which are shown to be a good approximation of the ℓ_2-minimal adversarial example for an input sample, and the assumption of flat decision boundaries around the constructed examples is believed to be practically valid [3, 82]. Therefore, we utilize it to generate $\mathbf{x}_\mathrm{p}$ and estimate $\mathbf{w}$ using the first order Taylor expansion of f at $\mathbf{x}_\mathrm{p}$ as:

$$\mathbf{w} = \nabla \mathrm{f}_{\mathrm{c}(\mathbf{x}_\mathrm{p})}(\mathbf{x}_\mathrm{p}) - \nabla \mathrm{f}_{\mathrm{c}(\mathbf{x})}(\mathbf{x}_\mathrm{p}). \tag{3.8}$$

In practice, the high-frequency structure of the gradients of DNNs increases the angle between $\mathbf{w}$ and $\widetilde{\mathbf{w}}$. Consequently, ρ in Equation 3.6 takes large values which often maps the input sample outside the legitimate range $[0, 1]^\mathrm{n}$. In the next section, we propose a smooth clipping technique to overcome this problem.

Algorithm 1 SmoothClip

```
1: input: Image $\mathbf{x}$, perturbation $\mathbf{r}$, low-pass filter $\mathbf{g}$, step size $\varepsilon$.
2: output: Smoothly clipped perturbation $\mathbf{r}_c$.
3: Initialize $\mathbf{r}_c \leftarrow \mathbf{r}$.
4: while $\max(\mathbf{x}+\mathbf{r}_c) > 1$ or $\min(\mathbf{x}+\mathbf{r}_c) < 0$ do
5:     $\mathbf{m}_0 = \mathbb{1}_{>0}(-(\mathbf{x}+\mathbf{r}_c)) * \mathbf{g}$,
6:     $\mathbf{m}_1 = \mathbb{1}_{>0}((\mathbf{x}+\mathbf{r}_c) - 1) * \mathbf{g}$,
7:     $\Delta_1 = \max(\mathbf{x}+\mathbf{r}_c - 1)\mathbf{m}_1$,
8:     $\Delta_0 = \min(\mathbf{x}+\mathbf{r}_c)\mathbf{m}_0$,
9:     $\mathbf{r}_c \leftarrow \mathbf{r}_c - \varepsilon(\Delta_1 + \Delta_0)$,
10: end while
11: return $\mathbf{r}_c$.
```

3.3.3 Validating Perturbations

The final adversarial example should reside inside the valid range of the input domain. An ordinary approach to hold this condition, especially in iterative attacks, is to clip the resulting adversarial examples [11, 82]. The clipping function, Clip, takes the constructed adversarial image and truncates each pixel value independently to fall within the valid range of the input space. However, applying this to smooth perturbations as: $\mathbf{r}_c = \text{Clip}(\mathbf{x}+\mathbf{r}) - \mathbf{x}$, will deteriorate the smoothness of perturbation. This is because the clipping function truncates each pixel individually and discards the local correlation between neighborhood perturbations. Specifically, this issue happens at edges and high-frequency areas of $\mathbf{x}$ as shown in Figure 3.5. A closed-form solution for smooth clipping, which should consider neighborhood correlation of perturbations (based on $\mathbf{g}$), results in a high complexity solution. We propose a simple and iterative approach for smoothly clipping the out-of-bound pixels. In the i^{th} iteration, when the range of $\mathbf{x}+\mathbf{r}^{\text{i}}$ remains out of the valid range, we compute masks $\mathbf{m}_0$ and $\mathbf{m}_1$ as indicators of pixels which exceed the valid bound as:

$$\mathbf{m}_0^{\text{i}} = \mathbb{1}_{>0}(-(\mathbf{x}+\mathbf{r}^{\text{i}})), \tag{3.9}$$

$$\mathbf{m}_1^{\text{i}} = \mathbb{1}_{>0}((\mathbf{x}+\mathbf{r}^{\text{i}}) - 1), \tag{3.10}$$

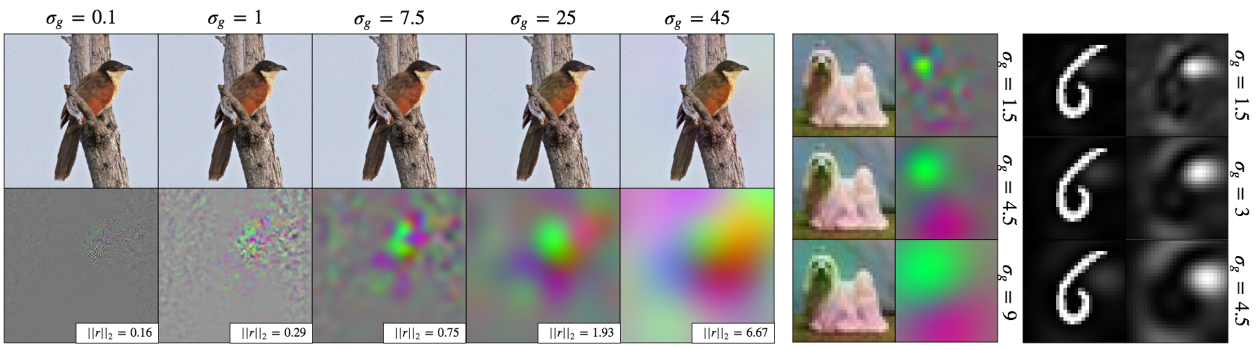


Figure 3.4: Visual demonstration of increasing smoothness of adversarial perturbations. Each set of images, from left to right, show adversarial examples and smooth adversarial perturbations computed for samples from ImageNet, CIFAR-10, and MNIST datasets on ResNet-101, ResNet-18, and LeNet architectures, respectively. Samples are from *'coucal'*, *'dog'*, and *'6'* classes and misclassified as *'robin'*, *'bird'*, and *'0'*.

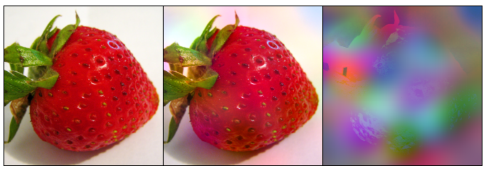


Figure 3.5: An example of applying a normal clipping on a smooth AP. Left: a benign sample correctly classified as *'strawberry'* by VGG16. Middle: an adversarial example classified as *'pineapple'*. Right: the perturbation after normal clipping.

where $\mathbb{1}_{>0}(.)$ is an indicator function that outputs 1 for elements greater than zero. To incorporate the neighborhood correlation of perturbations, we use the exact low-pass filter $\mathbf{g}$ used in Equation 3.7 to propagate the out-of-bound error to the neighborhood perturbations as: $\mathbf{m}_1^{\mathrm{i}} \leftarrow \mathbf{m}_1^{\mathrm{i}} * \mathbf{g}$ and $\mathbf{m}_0^{\mathrm{i}} \leftarrow \mathbf{m}_0^{\mathrm{i}} * \mathbf{g}$. Then, using a step size ε and maximum value of the out-of-bound error, we adjust the perturbation as:

$$\mathbf{r}^{\mathrm{i}+1} = \mathbf{r}^{\mathrm{i}} - \varepsilon \max(\mathbf{x} + \mathbf{r}^{\mathrm{i}} - 1)\mathbf{m}_1^{\mathrm{i}} - \varepsilon \min(\mathbf{x} + \mathbf{r}^{\mathrm{i}})\mathbf{m}_0^{\mathrm{i}}. \tag{3.11}$$

This iterative algorithm terminates when all pixels in $\mathbf{x} + \mathbf{r}^{\mathrm{i}}$ reside within the valid range. We refer to this algorithm as *SmoothClip*, and Algorithm 1 summarizes its functionality.

Algorithm 2 SmoothFool

```
1: input: Image $\mathbf{x}$, low-pass filter $\mathbf{g}$.
2: output: Smooth perturbation $\mathbf{r}$.
3: Initialize $\mathbf{x}^0 \leftarrow \mathbf{x}$, $\mathrm{i} \leftarrow 0$.
4: while $\mathrm{c_f}(\mathbf{x}^0) = \mathrm{c_f}(\mathbf{x}^\mathrm{i})$ do
5:     $\mathbf{r}_\mathrm{p} = \mathrm{DeepFool}(\mathbf{x}^\mathrm{i})$,
6:     $\mathbf{x}_\mathrm{p} = \mathbf{x}^\mathrm{i} + \mathbf{r}_\mathrm{p}$,
7:     $\mathbf{w}^\mathrm{i} = \nabla \mathrm{f}_{\mathrm{c}(\mathbf{x}_\mathrm{p})}(\mathbf{x}_\mathrm{p}) - \nabla \mathrm{f}_{\mathrm{c}(\mathbf{x})}(\mathbf{x}_\mathrm{p})$,
8:     $\widetilde{\mathbf{w}}^\mathrm{i} = \mathbf{g} * \mathbf{w}^\mathrm{i}$,
9:     $\mathbf{r}^\mathrm{i} = \frac{\mathbf{w}^{\mathrm{i}\top}(\mathbf{x}_\mathrm{p}^\mathrm{i} - \mathbf{x}^\mathrm{i})}{\mathbf{w}^{\mathrm{i}\top}\widetilde{\mathbf{w}}^\mathrm{i}}\widetilde{\mathbf{w}}^\mathrm{i}$,
10:    $\mathbf{r}^\mathrm{i} \leftarrow \mathrm{SmoothClip}(\mathbf{x}^\mathrm{i}, \mathbf{r}^\mathrm{i}, \mathbf{g})$,
11:    $\mathbf{x}^{\mathrm{i}+1} \leftarrow \mathbf{x}^\mathrm{i} + \mathbf{r}^\mathrm{i}$,
12:    $\mathrm{i} \leftarrow \mathrm{i} + 1$,
13: end while
14: return $\mathbf{x}^\mathrm{i} - \mathbf{x}^0$.
```

3.3.4 General Solution

In a general case for a non-linear classifier f, there is no guarantee that perturbations computed by Equation 3.7 cause input samples to pass the actual non-linear boundary. Figure 3.3 demonstrates a visualization of this fact. To overcome this problem, we adopt an iterative procedure. In each iteration, using the closest adversarial example, $\mathbf{x}_\mathrm{p}^\mathrm{i}$, corresponding to the sample $\mathbf{x}^\mathrm{i}$, we linearize the decision boundary and compute the smooth projection of $\mathbf{x}^\mathrm{i}$ on the approximated hyperplane using Equation 3.7.

Afterward, we smoothly rectify the resulting perturbation, $\mathbf{r}$, and repeat this procedure until $\mathrm{c}(\mathbf{x}^\mathrm{i}) \neq \mathrm{c}(\mathbf{x}^0)$, as detailed in Algorithm 2. Here, the smoothness of the final perturbation depends on the smoothness in each iteration. Consider $\mathrm{r^{tot}} = \mathrm{x^i} - \mathrm{x^0} = \sum_{\mathrm{j}=0}^{\mathrm{i}} \mathrm{r^j}$, where i is the total number of iterations, and $\mathrm{r^j}$ is the $\mathrm{j^{th}}$ smooth AP. It can be shown that the roughness of the overall perturbation is bounded as: $\boldsymbol{\Omega}(\mathrm{r^{tot}};\mathrm{h}) \leq \mathrm{i}^2 \max_\mathrm{j} \boldsymbol{\Omega}(\mathrm{r^j};\mathrm{h})$. To compute an AP with the desired level of roughness defined by h, we select g such that $\forall \mathrm{j} : \boldsymbol{\Omega}(\mathrm{r^j};\mathrm{h}) \ll ||\mathrm{r^j}||_2^2$, *i.e.*, the cut-off frequency of g should be smaller than h. In practice, $\max_\mathrm{j} \boldsymbol{\Omega}(\mathrm{r^j};\mathrm{h}) \ll \mathrm{i}^{-2}$, *i.e.*, even for a significantly smooth choices of $\mathbf{g}$ the algorithm converges in few iterations.

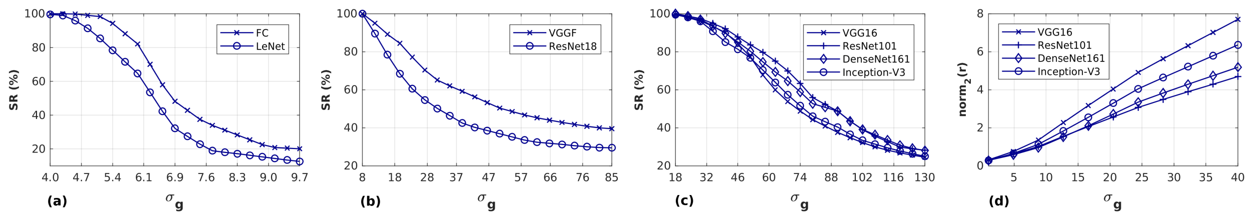


Figure 3.6: a, b, c) Fooling rate of the attack versus smoothing factor $\boldsymbol{\sigma_g}$ on MNIST, CIFAR-10 and ImageNet, respectively. d) Magnitude of perturbations vs. $\boldsymbol{\sigma_g}$ on ImageNet.

3.4 Experiments

3.4.1 Setup

We evaluate our attack on three datasets including the test set of MNIST [94], the test set of CIFAR-10 [95], and $10,000$ samples from the validation set of ILSVRC2012 [96] (10 images per each class). For the MNIST dataset, a two-layer fully-connected network (FC2) and a LeNet [97] architecture are used. For the CIFAR-10 dataset, we use a VGG-F [98] and ResNet-18 [12] architectures. For the ImageNet dataset, we consider VGG16 and ResNet-101. The accuracy of each model on benign samples is shown in Table 3.2. We set the step size $\boldsymbol{\varepsilon}$ of the SmoothClip to 1, 0.5, 0.1 for MNIST, CIFAR-10, and ImageNet respectively, which results in a fast and reasonable performance.

3.4.2 Definition of Smoothness

We define smoothness based on the Gaussian blur function since it is practical and the cut-off frequency can be easily modified by the standard deviation. We assume $\mathbf{h}$ and $\mathbf{g}$ to be Gaussian blur filters with isotropic standard deviations $\boldsymbol{\sigma_h}$ and $\boldsymbol{\sigma_g}$, respectively. In this setup, selecting any $\boldsymbol{\sigma_g} > \boldsymbol{\sigma_h}$ will minimize the roughness defined by $\mathbf{h}$. Increasing $\boldsymbol{\sigma_g}$ improves the smoothness of adversarial perturbations but reduces the performance of the attack. To implement the Gaussian kernel, we set the kernel width to $5\boldsymbol{\sigma}$.

3.4.3 Comparisons

For $\boldsymbol{\sigma}_{\mathbf{g}} \ll 1$, the proposed approach converges to DeepFool [3]. Hence, we use it as a baseline to compare magnitude of the generated perturbations. For the second baseline, we develop an attack based on [4] by replacing the classical TV loss term with the roughness penalty $\boldsymbol{\Omega}(\mathbf{r};\mathbf{h})$ from Equation 3.4 to provide a fair comparison framework as:

$$\arg\min_{\mathbf{r}} \Big(-\mathrm{J_c}(\mathrm{f}(\mathbf{x}+\mathbf{r}), \mathrm{y}_{\mathbf{x}}) + \lambda_{\mathrm{s}} \boldsymbol{\Omega}(\mathbf{r}; \boldsymbol{\sigma}_{\mathbf{h}}) \Big), \tag{3.12}$$

where $\mathrm{J_c}$ is the cross-entropy loss function, and $\mathrm{y}_{\mathbf{x}}$ denotes the ground truth label of sample $\mathbf{x}$. We refer to this method as the *iterative smooth* (IS) attack, and optimize it using gradient descent with a initial step size (learning rate) of 10^{-3}, and decay of 0.5 per each 100 iterations. We set λ_{s} to 0.1, 0.01 and 0.05 for MNIST, CIFAR-10, and ImageNet, respectively, which results in the most possible smooth perturbations for $\boldsymbol{\sigma}_{\mathbf{h}} = 1$. We consider this attack as the second baseline. We also compare the proposed method to the semantic adversarial examples given in [6], and refer to it as the *color-shift* (CS) attack, and consider it as the third baseline. We set the number of random trails of the CS algorithm to 100. Since CS adds perturbations in the HSV color space, we compute the average magnitude of perturbations for this attack in the HSV space to provide a fair comparison (the magnitude of adversarial perturbations in RGB color space is observed to be approximately 10 times greater).

3.4.4 Evaluation metrics

To compare results, We measure the fooling rate of the attack and average smoothness of constructed adversarial perturbations. The fooling rate is defined on the set of correctly classified benign samples since it provides a more robust measure to evaluate the attack. For the sake of brevity of explaining results, we define $\boldsymbol{\sigma}_{\mathbf{g}}^{\mathrm{A\%}}$ as the maximum value of $\boldsymbol{\sigma}_{\mathbf{g}}$ (minimum among network architectures) that results in a A% fooling rate. In order to evaluate the smoothness of constructed adversarial perturbations, we measure the expected roughness $\overline{\boldsymbol{\Omega}} = \mathbb{E}_{\mathscr{D}_{\mathrm{s}}}[\boldsymbol{\Omega}(\mathbf{r}_{\mathbf{x}}, \mathbf{h})]$, where $\mathbf{r}_{\mathrm{x}}$ is the AP constructed for $\mathbf{x}$, and $\mathscr{D}_{\mathrm{s}}$ is the set of successfully attacked samples. This measure is sensitive to the magnitude of perturbations. Thus, we develop a second measure by normalizing $\boldsymbol{\Omega}$ over the total power of perturbations

		Execution time (sec.)					
Dataset	**Net.**	DF	IS	CS	SF_1	SF_2	SF_3
MNIST	1	0.02	1.43	-	0.03	0.07	0.43
	2	0.02	1.94	-	0.03	0.13	0.47
CIFAR-10	3	0.06	8.25	0.03	0.07	0.11	0.38
	4	0.12	10.43	0.06	0.13	0.16	0.51
ImageNet	5	0.41	24.53	0.19	0.55	0.59	1.13
	6	0.87	29.47	0.24	0.94	1.22	1.49

Table 3.1: Comparing the execution time of the algorithm to other attacks. SF_1, SF_2 and SF_3 are SmoothFool with different smoothing levels $\boldsymbol{\sigma}_{g_1}$, $\boldsymbol{\sigma}_{g_2}$ and $\boldsymbol{\sigma}_{g_3}$ which are set to be $\{1,3,5\}$, $\{1,5,10\}$, and $\{1,10,20\}$ for MNIST, CIFAR-10 and ImageNet, respectively. Numbers in the second column are associated with corresponding network architectures in Table 3.2.

as: $\overline{\boldsymbol{\Omega}}_{\mathrm{n}} = \mathbb{E}_{\mathscr{D}_{\mathrm{s}}}[\boldsymbol{\Omega}(\mathbf{r}_{\mathbf{x}}, \mathbf{h})/||\mathbf{r}_{\mathbf{x}}||_2^2]$. Indeed, $\overline{\boldsymbol{\Omega}}_{\mathrm{n}}$ relatively measures how much of the total power of the adversarial perturbations is occupied by high-frequency components according to $\mathbf{h}$.

Dataset	**Network**	**Acc.**	**F.Rate** (%)		$\overline{\Omega} \times 10^3 @ \sigma_h = 1$				$\overline{\Omega}_n \times 10^3 @ \sigma_h = 1$				$\mathbb{E}_x[\|\|r_x\|\|_2] @ \sigma_g$			
		(%)	**IS**	**CS**	**DF**	**IS**	**CS**	**SF**	**DF**	**IS**	**CS**	**SF**	**DF**	**IS**	**CS**	**SF**
MNIST	1-FC2	98.6	98.0	-	783	591	-	**334**	511	308	-	**63**	1.17	2.81	-	1.76
	2-LeNet	99.1	94.1	-	890	677	-	**352**	532	338	-	**68**	1.23	3.18	-	2.32
CIFAR-10	3-VGG-F	93.1	85.4	93.4	288	203	184	**114**	891	216	163	**57**	0.19	3.14	4.42	1.60
	4-ResNet-18	93.3	87.8	89.1	310	206	199	**127**	959	287	175	**65**	0.21	3.63	4.93	1.60
ImageNet	5-VGG16	71.5	57.8	91.1	111	89	104	**29**	871	358	238	**51**	0.25	4.90	7.71	0.58
	6-ResNet-101	77.3	62.6	92.7	108	83	95	**16**	827	300	207	**36**	0.28	4.82	7.56	0.55

Table 3.2: Comparing SmoothFool (SF) to DeepFool (DF) [3], iterative smooth (IS) [4, 5], and color-shift (CS) [6] attacks. To satisfy smoothness based on $\boldsymbol{\sigma_h}=1$, $\boldsymbol{\sigma_g}$ is set to 2 for all datasets. Fooling rates of SF and DF are $>99.9\%$ on all datasets.

.

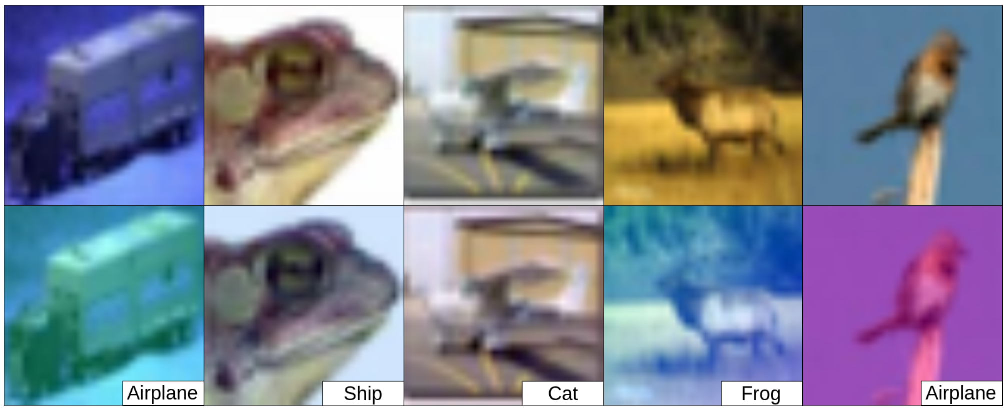


Figure 3.7: Examples of extremely smooth adversarial perturbations computed for ResNet-18 and CIFAR-10 dataset with $\boldsymbol{\sigma}_\mathbf{g} = 200$.

3.4.5 General Performance

Figure 3.6 (a-c) shows the fooling rates of SmoothFool versus $\boldsymbol{\sigma}_\mathbf{g}$. We observe that the pair $(\boldsymbol{\sigma}_\mathbf{g}^{100\%}, \boldsymbol{\sigma}_\mathbf{g}^{20\%})$ for MNIST, CIFAR-10 and ImageNet is $(4.1, 7.8)$, $(8.4, 124.4)$ and $(19.3, 165.3)$, respectively. As expected, the fooling rate is highly dependent on the smoothing factor $\boldsymbol{\sigma}_\mathbf{g}$. However, the fooling rate remains high for significantly large (compared to the size of the input image) values of $\boldsymbol{\sigma}_\mathbf{g}$ on ImageNet and CIFAR-10. For instance, $\boldsymbol{\sigma}_\mathbf{g}^{50\%}$ for CIFAR-10 is 32.8 which is approximately equal to the width of input images and shows that it is possible to fool the classifier on 50% of samples solely by adding a carefully selected constant value to all pixels of each color channel. The magnitude of smooth adversarial perturbations versus smoothness is depicted in Figure 3.6 (d). Increasing smoothness results in larger magnitudes of adversarial perturbations since the projection of $\widetilde{\mathbf{w}}$ onto $\mathbf{w}$ will become smaller. However, smoothness of perturbations allows larger magnitudes since they are not as perceptible when compared to the noisy structure of contemporary adversarial perturbations.

We observe in Table 3.2 that SmoothFool with $\boldsymbol{\sigma}_\mathbf{g} = 2$ on all datasets, crafts significantly smoother (based on $\overline{\boldsymbol{\Omega}}$ and $\overline{\boldsymbol{\Omega}}_\mathrm{n}$ with $\boldsymbol{\sigma}_\mathbf{h} = 1$) adversarial perturbations compared to the baseline attacks for the smoothness, while the magnitudes of adversarial perturbations are solely 1.8x larger than the SOTA ℓ_2-minimal adversarial perturbations crafted by DeepFool. In addition, the execution time of the algorithm detailed in Table 3.1 shows that the proposed method computes adversarial perturbations (with fooling rate $> 99\%$) at least 20x faster than the

Class	**VGG-F** σ_g 20	60	100	**ResNet-18** σ_g 20	60	100
airplane	75.0	29.5	25.0	67.3	34.7	28.2
automobile	**58.7**	**10.8**	**6.5**	**33.3**	**6.6**	**2.2**
bird	93.4	65.2	<u>52.1</u>	65.1	41.8	32.5
cat	<u>100</u>	58.3	43.7	65.3	21.1	17.3
deer	87.2	<u>60.0</u>	49.0	78.0	40.2	36.0
dog	78.7	48.9	40.4	75.1	<u>55.5</u>	<u>52.7</u>
frog	88.8	51.1	46.6	68.9	44.8	41.3
horse	74.5	29.0	23.6	70.3	25.9	22.2
ship	79.0	32.5	25.5	<u>81.4</u>	35.1	29.6
truck	73.9	32.6	21.7	68.5	31.4	22.8
all	83.8	45.8	37.6	67.2	33.0	27.9

Table 3.3: Per-class fooling rate (%) of SmoothFool for three values of σ_g on the CIFAR-10 dataset. Bold and underlined values show the fooling rate on classes with highest and lowest robustness against smooth adversarial perturbations, respectively.

	Fooling rate under defense (%)					
Defense	IGSM	DF	CS	SF_1	SF_2	SF_3
Adv.	32.6	15.6	64.5	58.6	70.7	**78.0**
PGD	21.0	12.3	61.4	57.2	67.3	**72.8**
Ens.	18.7	14.0	62.2	54.5	62.8	**73.6**
SAT	22.8	37.2	21.0	11.5	42.9	**53.4**
HGD	9.3	11.2	46.9	43.7	57.2	**66.2**
MagNet	10.7	8.9	25.1	46.4	**65.5**	52.6

Table 3.4: Evaluating attacks under different defense strategies on a ResNet-18 trained on CIFAR-10. SF_1, SF_2, and SF_3 denote the proposed algorithm with $\boldsymbol{\sigma_g}$ of 1, 3, and 5, respectively.

IS method (with fooling rate $\simeq 65\%$ on ImageNet). Figure 3.4 shows some examples of smooth adversarial perturbations computed for different levels of smoothness. We observe that each class responds differently as the smoothness of adversarial perturbations increases. Table 3.3 shows the per-class fooling rate of the attack on CIFAR-10. Smooth perturbations at $\boldsymbol{\sigma_g} = 100$ fool the VGG-F classifier on more than 50% of samples of the *'bird'* class, while they are approximately not effective for the *'car'* class. This shows that some classes are severely sensitive to smooth perturbations while other exhibit lower sensitivity. The network architecture has a direct effect on this observation since the most sensitive class to smooth adversarial perturbations for each specific value of $\boldsymbol{\sigma_g}$ is different among network architectures.

Figure 3.7 demonstrates some examples of extremely smooth adversarial perturbations on CIFAR-10, showing a similar behavior (in RGB color space) as color-shifted adversarial examples [6]. However, as the method in [6] randomly shifts Hue and Saturation of benign samples, it often generates odd adversarial examples such as *'blue apples'* or *'red lemons'* which are no longer adversarial examples since the conceptual evidence of objects is destroyed. However, since SmoothFool finds relatively small smooth perturbations, the whole concept of an object will not change drastically after the attack.

Performance under white-box defenses. Here, we evaluate the effectiveness of smooth perturbations against defense methods. First, we evaluate the attack under defenses based on adversarial training on FGSM (Adv.) [10], projectile gradient descent (PGD) [86] and ensemble (Ens.) [76] adversarial examples. We consider an additional defense of training on adversarial examples computed by the proposed SmoothFool with $\boldsymbol{\sigma}_\mathbf{g} = 1$, and refer to it as Smooth Adversarial Training (SAT). Second, we consider the high-level guided denoiser (HGD) [50] as a denoising based defense and MagNet [92] as a defense which evaluates adversarial examples using a learned distribution of natural samples. In all experiments, we assume that attacks have zero knowledge about the defense models.

Table 3.4 shows the performance of SmoothFool under defenses. Results suggest that increasing the smoothness of adversarial perturbations elevates the chance of bypassing defense methods. Such a characteristic had been observed before in adversarial examples constructed by spatial transformations [1, 38]. Smooth adversarial perturbations with $\boldsymbol{\sigma}_\mathbf{g} = 5$ successfully bypass HGD defense and defenses based on adversarial training on more than 60% of samples. Similarly, the CS attack shows significant robustness against all defenses except MagNet. A reasonable explanation is that although the CS attack generates relatively smooth adversarial perturbations compared to conventional attacks, changing the Hue and Saturation of images considerably pushes samples outside the distribution of natural samples leaned by MagNet. SmoothFool bypasses MagNet by a notable margin which indicates the closeness of generated samples to the distribution of natural images. However, for large values of $\boldsymbol{\sigma}_\mathbf{g}$, the magnitude of smooth adversarial perturbations takes large values, and thus, degrades the fooling rate of SmoothFool against MagNet defense. Furthermore, we observe that SAT defense provides a relative robustness against smooth adversarial perturbations constructed by $\boldsymbol{\sigma}_\mathbf{g} = 1$, but is susceptible to smoother perturbations. This suggest that the frequency components of adversarial perturbations can play a crucial role in bypassing adversarial training defenses trained on examples constructed by adversarial perturbations of different frequency components.

Black-box performance and ablation on smoothing functions. Here, we evaluate the black-box performance of smooth adversarial perturbations. Since our algorithm computes ℓ_2-minimal perturbations, we scale smooth adversarial perturbations to have the maximum

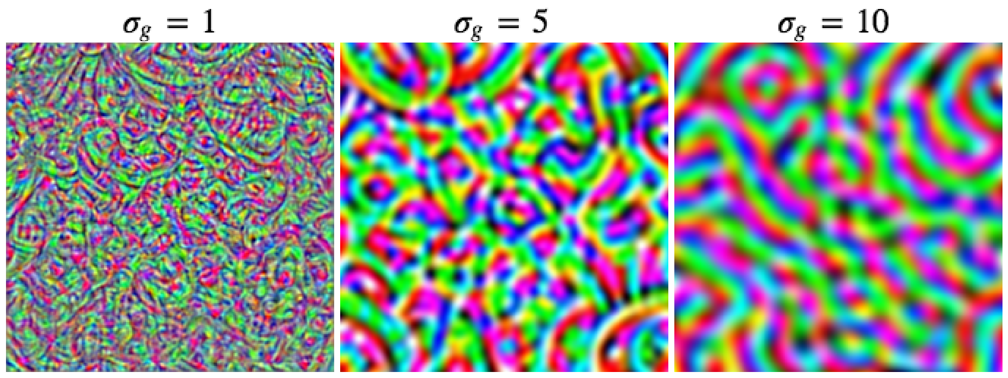


Figure 3.8: Smooth universal adversarial perturbations crafted for VGG16 architecture (best viewed in color).

ℓ_∞-norm of 16 for pixel values in range 255 based on the conventional setting for black-box attacks on ImageNet [88]. We consider two additional smooth functions including linear and uniform kernels to evaluate the effect of smoothing functions on fooling rates and transferability of adversarial perturbations. The uniform kernel of size k has all values equal to $\frac{1}{\mathrm{k}^2}$. The linear kernel of size k has the maximum value of $\frac{4}{\mathrm{k}^2}$ at the center and minimum value of zero at edges. Other values are the linear interpolation of the min. and max. values.

Table 3.5 presents the results for this experiment. The fooling rate of attacks is 100% when the source and target models are the same. This suggests that the type of smoothing functions does not constrain the performance of adversarial perturbations. Hence, a broad range of smoothing functions can be deployed for generating smooth adversarial perturbations. Transferability of adversarial examples consistently improves as the smoothness of perturbations increases. This demonstrates that smoothness increases the transferability of adversarial examples for black-box attacks which validates the results reported by Dong et al. [88].

Universal adversarial perturbations.

We integrate the proposed approach with the universal adversarial perturbations (UAP) [99] to explore the possibility of finding smooth UAPs. The implementation detail and the integrated algorithm is available in the Supplementary. We compute smooth UAPs for VGG16 and then evaluate their transferability on four networks including ResNet-101, ResNet-151, DenseNet-161, and Inception-V3. Table 3.6 demonstrates the performance of smooth UAPs

Net.	Smoothing	Param.	VGG16	ResNet101	Inc-V3
VGG16	-	-	100	12.6	8.9
	Gaussian	$\sigma = 5$	100	15.8	13.6
		$\sigma = 10$	100	20.7	15.3
	Linear	k = 25	100	17.7	11.6
		k = 50	100	23.5	14.1
	Uniform	k = 25	100	16.8	12.0
		k = 50	100	21.8	14.6
ResNet101	-	-	15.2	100	15.0
	Gaussian	$\sigma = 5$	17.0	100	18.8
		$\sigma = 10$	19.9	100	22.5
	Linear	k = 25	19.8	100	16.9
		k = 50	22.8	100	19.7
	Uniform	k = 25	18.6	100	17.3
		k = 50	21.0	100	22.1

Table 3.5: Transferability of smooth perturbations for black-box attack. Columns show source networks and attack parameters, and rows show the target models.

σ_g	VGG16	RNet101	RNet152	DNet161	Inc-V3
0	78.3	64.8	63.4	52.9	54.6
1	79.6	66.0	66.8	53.2	57.8
5	82.2	**69.9**	**70.3**	**57.6**	58.6
10	**84.5**	68.7	69.1	55.9	**61.6**

Table 3.6: Transferability of universal smooth adversarial perturbations computed for VGG16 accross data points and network architectures.

versus smoothness. Increasing smoothness enhances the transferability of adversarial perturbations across both the data points and network architectures. The transferability on 3 networks deteriorates for $\sigma_g > 5$. We attribute this observation to the theoretical fact that increasing smoothness also increases the magnitude of adversarial perturbations. Hence, with the same threshold for the maximum ℓ_∞-norm of smooth Uadversarial perturbations, there always exist a σ_g after which the transferability decreases.

3.5 Conclusion

In this study, we explored the vulnerability extent of DNNs to smooth adversarial perturbations by proposing SmoothFool, a framework for computing ℓ_2-minimal smooth adversarial perturbations. The methodology is developed based on a broad definition of smoothness and can be extended to pose any frequency-domain constraint on perturbations. Through extensive experiments, we validated the effectiveness of smooth adversarial perturbations on deep classifiers robustified by two major group of defense strategies. Smoothness of perturbations improves the transferability of adversarial examples across network architectures and data points. Furthermore, we observed that class categories exhibit variable susceptibility to smooth perturbations. Our results suggest that adversarial perturbations with modified frequency-domain characteristics can provide a new and powerful tool for evaluating the adversarial vulnerability.

Chapter 4

Exploiting Joint Robustness to Adversarial Perturbations

4.1 Introduction

DNNs have played an astonishing role in the evolution of modern machine learning by achieving SOTA performance on many challenging tasks [18, 100]. Despite their excellent performance, scalability, and generalization to unseen test data, they suffer from a major drawback: slight manipulations of the input samples can form *adversarial examples* causing drastic changes in the predictions of the model [3,10,32]. Perturbations required for this aim are often quasi-imperceptible to the human eye and can transfer across classifiers [56, 58], data samples [99, 101], and input transformations [102, 103]. This issue has raised increasing concerns regarding the deployment of DNNs in security-sensitive applications such as autonomous vehicles, biometric identification, and e-commerce.

Initially, a large body of work has been devoted to addressing the problem by heuristic approaches built upon the empirically observed characteristics of perturbations, such as their noisy structure. However, the uncertainty of assumptions and lack of formal explanations for the phenomenon has caused the majority of the defense attempts to be compromised by more advanced attacks [38, 58, 104]. Recent studies have made significant progress in explaining the cause of adversarial vulnerability by demonstrating that adversarial examples are natural consequences of non-zero test error of classifiers in the data space [105, 106].

Particularly, due to the huge cardinality of the input space, a small number of misclassified points around a natural input sample forms a very close decision boundary which can be reached by adversarial perturbations. This suggests that adversarial robustness can only be certified for bounded perturbations [40, 106] since achieving zero error rate is nontrivial in general [105].

The majority of studies on adversarial robustness have concerned single classifiers [10, 32, 40, 86, 105, 106]. However, exploring interactions of multiple classifiers has highlighted the potential of ensembles for mitigating the adversarial vulnerability [107–110]. In this paper, we exploit first-order interactions in ensembles to provably improve the robustness of the ensemble prediction. We illustrate that the diversity of the gradient directions and the balance of the gradient magnitudes are two key factors for enhancing the robustness of deep ensembles. Specifically, we make the following contributions: i) We introduce a practically feasible case of interactions within ensembles which is certified to improve the robustness of the model against *white-box* attacks. ii) We propose a training framework termed joint gradient phase and magnitude regularization (GPMR) to impose the desired interactions among the members of the ensemble. iii) We validate the effectiveness of the proposed method using extensive experiments including gradient-based and gradient-free evaluations. iv) We demonstrate that the proposed training framework is orthogonal to previous approaches that aim to provide adversarial robustness by bounding the magnitude of the gradients, such as adversarial training.

4.2 Related Work

4.2.1 Methods for Crafting Adversarial Examples

Despite the highly non-linear nature of DNNs, they have been observed to exhibit linear characteristics around the actual parameters of the model and the input samples [10, 89, 90]. In particular, Goodfellow et al. [10] showed that the prediction of DNNs can be changed drastically by translating the input sample toward the gradient of the classification loss. Hence, they proposed the fast gradient sign method (FGSM) as a single step attack incorporating solely the sign of gradients to craft APs. Kurakin et al. [11] improved the performance of FGSM by adopting an iterative procedure called IGSM. Moosavi et al. [3] proposed DeepFool to find approximately ℓ_p-minimal adversarial perturbations by iteratively translating input samples toward the linearized approximation of the closest decision boundary. Our methodology builds on DeepFool to find minimal smooth APs. For more detailed description of the adversarial attacks please refer to [111].

4.2.2 Defending against Adversarial Examples

Myriad of studies have attempted to robustify DNNs employing approaches such as knowledge distillation [112], manifold learning [87,92], data transformation and compression [113, 114], statistical analysis [115], and regularization [111]. However, the majority of the defense schemes in the literature are compromised by more sophisticated attacks [58, 116]. An effective approach for improving the robustness of DNNs is *adversarial training* in which the training set is augmented by adversarial examples crafted during the training process. This approach is widely studied using different types of adversarial examples [3, 10, 11, 32, 86]. A major limitation of adversarial training is its dependence on the type of adversarial examples used for training the model. Thus, this approach cannot provide reliable robustness against unseen adversarial examples and out-of-distribution samples, e.g. crafted by additive Gaussian noise [105].

A group of studies proposed to directly limit the variation of predictions against slight input changes by bounding the Lipschitz constant of networks [40,117,118]. However, controlling the Lipschitz constant involves incorporating highly non-linear and intractable losses to the training objective, which results in restrictive computational costs for large-scale DNNs. Besides, theoretical assumptions for regularizing the Lipschitz constant of DNNs reduces the effectiveness of these approaches against strong attacks [119].

Another body of work has considered interactions of multiple classifiers to alleviate adversarial vulnerability [107–110]. The majority of these approaches propose a method to promote the diversity of predictions. Abbasi et al. [109] demonstrated that specializing members of the ensemble on different subsets of classes can provide robustness against adversarial examples. Bagnall et al. [110] proposed a joint optimization scheme to minimize the similarity of the classification scores on adversarial examples. Pang et al. [107] developed the adaptive diversity promoting (ADP) approach which diversifies the non-maximum predictions to maintain the accuracy of the model on natural examples.

However, diversifying the predictions does not provide reliable robustness in the *white-box* defense scenario, where all the parameters of the model are known by the adversary. In this setup, the adversary can use the gradients of diversified predictions to fool all classifiers at the same time. Moreover, we both theoretically and experimentally demonstrate that diversifying predictions does not improve the robustness in gradient-free evaluations since the gradient of classifiers can share similar directions. Recently, Kariyappa and Qureshi [108] considered the diversity of gradients in ensembles to provide adversarial robustness and proposed the gradient alignment loss (GAL).

However, this approach suffers from two limitations. First, gradient alignment loss (GAL) does not consider the optimal geometrical bounds for diversifying the gradient directions. This degrades the performance of the approach and causes significant fluctuations in the training process as discussed in section 4.4. Second, GAL does not equalize the magnitude of gradients of the members. Therefore, it is solely evaluated in the *black-box* threat model, where the attacker has no access to the model parameters or gradients. In the *white-box* attack scenario, the attacker can easily fool a few classifiers in the set which have the maximum gradient magnitudes at the input sample. In contrast, our work establishes a new theoretical framework for analyzing the joint robustness by finding the optimal first-order defensive interactions between the members of the ensemble in the white-box threat model.

4.3 Method

Altering the prediction of the classifier primarily changes the score of the predicted class. Therefore, in our theoretical analysis, we focus on the change of the final output of a differentiable classifier rather than the change in the index of the maximum argument of the output, *i.e.*, the predicted class. Considering ℓ_{p}-norm as the distance metric to measure the magnitude of perturbations, we define the robustness to adversarial examples or, more specifically, adversarial perturbations as follows:

Definition A function $\mathrm{f} : \mathcal{X} \subset \mathbb{R}^{\mathrm{n}} \to \mathbb{R}^{\mathrm{m}}$ is said to be $(\varepsilon, \delta)_{\mathrm{p}}$ robust to adversarial perturbations over the set $\mathcal{X}$, if for all samples $\mathrm{x}, \mathrm{x}' \in \mathcal{X}$ and $||\mathrm{x}-\mathrm{x}'||_{\mathrm{p}} \leq \varepsilon$, f satisfies $||\mathrm{f}(\mathrm{x})-\mathrm{f}(\mathrm{x}')||_2 \leq \delta$.

To compare the robustness of different classification schemes, we analyze the ℓ_{p}-norm lower bound of the magnitude of perturbations, ε, that is needed to change the maximum prediction by a fixed δ.

We analyze the robustness of a single classifier in Section 4.3.1. In Section 4.3.2, we formalize the robustness of ensembles for the case where the adversary has to change the prediction of all members to fool the ensemble prediction. Here, we introduce a practically feasible scenario, *i.e.*, a set of conditions, for interactions between the members of the ensemble and prove that it enhances the robustness of the ensemble. Afterward in Section 4.3.3, we adopt the proposed scenario for the practical threat models in which fooling a subset of classifiers in the ensemble is sufficient to change the ensemble prediction. Finally, we present our approach for imposing the desired defensive interactions in Section 4.3.4.

4.3.1 Robustness of a Single Classifier

Let $\mathrm{f} : \mathcal{X} \to \mathbb{R}^{\mathrm{m}}$ be a differentiable classifier mapping data point $\mathrm{x} \in \mathcal{X}$ to m classification scores $\mathrm{f}_{\mathrm{j}}(\mathrm{x})$, $\mathrm{j} \in \{0, \ldots, \mathrm{m}-1\}$. The true label for sample x is y, and the class predicted by the network is denoted by $\mathrm{c} = \arg\max_{\mathrm{j}} \mathrm{f}_{\mathrm{j}}(\mathrm{x})$. Any attempt to change the prediction of the network by translating the input sample x using perturbation r changes $\mathrm{f}_{\mathrm{c}}(\mathrm{x})$. We develop our methodology based on the first-order approximation of $\mathrm{f}_{\mathrm{c}}(\mathrm{x})$: $\mathrm{f}_{\mathrm{c}}(\mathrm{x}+\mathrm{r})-\mathrm{f}_{\mathrm{c}}(\mathrm{x}) \approx \langle \nabla_{\mathrm{x}} \mathrm{f}_{\mathrm{c}}, \mathrm{r} \rangle$* since DNNs exhibit linear characteristics around the input samples [10, 89, 90] where we seek to enhance the robustness. The minimal ℓ_{p}-norm perturbation $\mathrm{r}^*_{\mathrm{p}}$, for $\mathrm{p} \in [1, \infty)$, required to change the classification score by δ can be computed using the Hölder inequality and ℓ_{p}-norm projection [120, 121] as:

$$\mathrm{r}^* \approx \frac{\delta}{||\nabla \mathrm{f}_{\mathrm{c}}||_{\mathrm{q}}} \partial(||\nabla \mathrm{f}_{\mathrm{c}}||_{\mathrm{q}}), \tag{4.1}$$

*We drop x from the gradient operator for the rest of the paper.

where ℓ_p-norm and ℓ_q-norm are dual norms ($\frac{1}{p} + \frac{1}{q} = 1$), and $\partial(.)$ denotes the subgradient of the argument. For a differentiable ℓ_q-norm at ∇f_c, the subgradient is equal to: $\partial||\nabla f_c||_q/\partial\nabla f_c$, and Equation 4.1 can be rewritten as:

$$r^* \approx \left(\frac{\delta}{||\nabla f_c||_q}\right)\left(\frac{|\nabla f_c|^{q-1} \odot \text{sign}(\nabla f_c)}{||\nabla f_c||_q^{q-1}}\right), \tag{4.2}$$

where $\odot$ denotes element-wise multiplication. Similar first-order approximation of the lower bound of the ℓ_p-norm robustness has been previously derived [3,40]. Equation 4.2 implies that the magnitude of the gradient plays a crucial role in the robustness of the classifier. Hence, significant efforts have been made to directly smooth out f_c by controlling the Lipschitz constant [40, 117, 118] or adversarial training [10, 86]. Here, we take an orthogonal approach to the previous studies and seek to increase the lower bound of Equation 4.2 by exploring the joint robustness of multiple classifiers.

4.3.2 Joint Robustness of Multiple Classifiers

Let $\mathcal{F}$ be an ensemble of k classifiers, $\mathcal{F}=\{f^i\}_{i=0}^{k-1}$, where $f^i : \mathcal{X} \to \mathbb{R}^m$ maps the data point $x \in \mathcal{X}$ to m classification scores $f^i_j(x)$, $j \in \{0, \ldots, m-1\}$. The class predicted for x by the classifier $f^i \in \mathcal{F}$ is denoted by $c_i = \arg\max_j f^i_j(x)$. Following the previous studies on the robustness of ensembles [107–110], we assume that the ensemble prediction is the average of the prediction of individual classifiers as: $\mathcal{F}(x) = \frac{1}{k}\sum_{f\in\mathcal{F}} f(x)$, and the predicted class by the ensemble is: $c = \arg\max_j \mathcal{F}_j(x)$, where $\mathcal{F}_j$ is the predicted probability associated with the jth class. In this section, we relax the problem by assuming that the adversary has to fool all members in order to fool the ensemble prediction, *i.e.*, the ensemble rejects the input sample when $\exists\ i, j : c_i \neq c_j$.

The ℓ_p-norm minimal perturbation r^*_p required to decrease the classification scores of all classifiers at x by at least $\delta > 0$ is the solution of the following optimization problem:

$$\min ||r||_p \ \ \text{s.t.} \ \ \langle\nabla f^i_{c_i}, r\rangle \leq -\delta, \quad \forall i \in \{0, \ldots, k-1\}. \tag{4.3}$$

This interprets that the joint robustness within the ensemble is associated with the gradients of each individual member. Analyzing such interactions rely on the solution of this optimization problem which does not have an analytic form for the general ℓ_p-norm case but

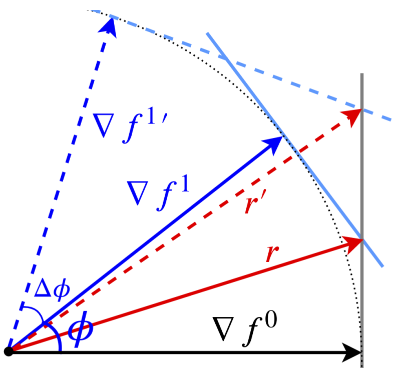


Figure 4.1: An illustration of Theorem 4.3.1 for $k = 2$. Increasing the angle between gradients ∇f^0 and ∇f^1 by $\Delta\phi$ increases the magnitude of the minimum perturbation from $||r||_2 = \frac{\sqrt{2}}{l\sqrt{\cos\phi+1}}$ to $||r'||_2 = \frac{\sqrt{2}}{l\sqrt{\cos(\phi+\Delta\phi)+1}}$.

can be computed using non-linear programming methods [122]. However, the ℓ_2-norm case when gradient vectors are linearly independent has the following closed-form solution:

$$r_2^* = -\delta\Omega^T(\Omega\,\Omega^T)^{-1}\mathbb{1}_{k\times 1}, \tag{4.4}$$

where $\Omega := [\nabla f^0_{c_0}, \nabla f^1_{c_1}, \dots, \nabla f^{k-1}_{c_{k-1}}]^T$ and $\mathbb{1}_{k\times 1}$ is an all-one matrix of size $k \times 1$.

The worst-case scenario for the joint robustness of k classifiers occurs when gradients of classifiers, $\nabla f^i_{c_i}$ for any given sample x, share the same direction. For this case, the magnitude of the optimal ℓ_2-norm solution for Equation 4.3 is: $||r_2^*||_2 \approx \frac{\delta}{\max_i\{||\nabla f^i_{c_i}||_2\}}$. Therefore, the ℓ_2-norm joint robustness offered by k classifiers in the worst-case scenario is of the same order as the robustness of a single classifier depicted in Equation 4.2.

Analyzing the characteristics of the optimal perturbation, r_2^*, for the multiple classifier framework is not analytically possible without considering additional constraints. In Theorem 4.3.1, we assume that the gradient vectors of all classifiers have an equal magnitude at each x, and they are equiangular, *i.e.*, the angle of any two gradient vectors is equal to ϕ. Hence, we derive a lower bound for the joint robustness of k classifiers with equiangular gradients.

Theorem 4.3.1. *Let* $\nabla f^0, \dots, \nabla f^{k-1}$ *be* k *vectors in* $\mathbb{R}^n$ *with an equal length* l, *and for any* $i \neq j \in \{0, \dots, k-1\}$, $\langle \nabla f^i, \nabla f^j\rangle = l^2\cos\phi$, *and let* $r \in \mathbb{R}^n$ *be a vector such that* $|\langle \nabla f^i, r\rangle| \geq |\delta|$

holds for any i, *then:*

$$||\mathrm{r}||_2 \geq \frac{|\delta|\sqrt{\mathrm{k}}}{\mathrm{l}\sqrt{((\mathrm{k}-1)\cos\phi + 1)}}. \tag{4.5}$$

Proof. Rewriting Equation 4.4 gives the minimal ℓ_2-norm solution, r, satisfying $|\langle \nabla \mathrm{f}^{\mathrm{i}}, \mathrm{r}\rangle| \geq |\delta|$ as: $\mathrm{r}_2^* = \sum_{\mathrm{i}} \alpha_{\mathrm{i}} \nabla \mathrm{f}^{\mathrm{i}}$, where α_{i} is the i^{th} element of the vector $\alpha = \delta(\Omega\ \Omega^{\mathrm{T}})^{-1}\mathbb{1}_{\mathrm{k}\times 1}$, and $\Omega = [\nabla \mathrm{f}^0, \ldots, \nabla \mathrm{f}^{\mathrm{k}-1}]^{\mathrm{T}}$. Applying the equiangular condition, we have: $\alpha_{\mathrm{i}} = \dfrac{\delta}{\mathrm{l}^2((\mathrm{k}-1)\cos\phi + 1)}$, which is independent of i. On the other hand, $||\sum_{\mathrm{i}=0}^{\mathrm{k}-1} \nabla \mathrm{f}^{\mathrm{i}}||_2^2 = \langle \sum_{\mathrm{i}=0}^{\mathrm{k}-1} \nabla \mathrm{f}^{\mathrm{i}}, \sum_{\mathrm{i}=0}^{\mathrm{k}-1} \nabla \mathrm{f}^{\mathrm{i}}\rangle = \mathrm{kl}^2((\mathrm{k}-1)\cos\phi + 1)$. Combining these two equations concludes the proof. □

For the equiangular case, when k classifiers are identical, *i.e.*, $\phi = 0$, members of the ensemble have the minimum defensive interactions since the joint robustness is equal to the robustness of a single classifier obtained in Equation 4.2. For k classifiers with orthogonal gradient vectors, the lower bound is equal to $||\mathrm{r}||_2 = \delta\frac{\sqrt{\mathrm{k}}}{\mathrm{l}}$ and the robustness is of $\mathrm{o}(\frac{\sqrt{\mathrm{k}}}{\mathrm{l}})$. As ϕ grows, the robustness increases and approaches infinity when $\phi \rightarrow \arccos(\frac{-1}{\mathrm{k}-1})$. The robustness for an arbitrary set of gradients, $\{\nabla \mathrm{f}^0, \ldots, \nabla \mathrm{f}^{\mathrm{k}-1}\}$, is lower bounded by the robustness of any set of inscribed equiangular vectors with $\phi = \min_{\mathrm{i}\neq \mathrm{j}} \angle(\nabla \mathrm{f}^{\mathrm{i}}, \nabla \mathrm{f}^{\mathrm{j}})$ and $\mathrm{l} = \max_{\mathrm{i}} ||\nabla \mathrm{f}^{\mathrm{i}}||_2$. Therefore, Theorem 4.3.1 provides a lower bound to the robustness of the general case of the gradients. This implies that the robustness of ensembles can be improved by increasing the minimum angle between gradients and decreasing the maximum gradient magnitude. Figure 4.1 illustrates how promoting the gradient diversity improves the robustness.

Diversity of the gradient directions has been studied before in GAL [108] as a heuristic methodology to improve the robustness against *black-box* attacks. Theorem 4.3.1 highlights two shortcomings of GAL limiting its effectiveness against *white-box* attacks. First, GAL does not consider the optimal bound $\arccos(\frac{-1}{\mathrm{k}-1})$ for the gradient diversity. We observe that this causes a fluctuation in the training of GAL and reduces the effectiveness of diversifying the gradient directions. Second, GAL does not regularize the gradient magnitudes among members. Consequently, any *white-box* attack to the ensemble prediction can easily circumvent the defensive strategy by targeting the least robust members.

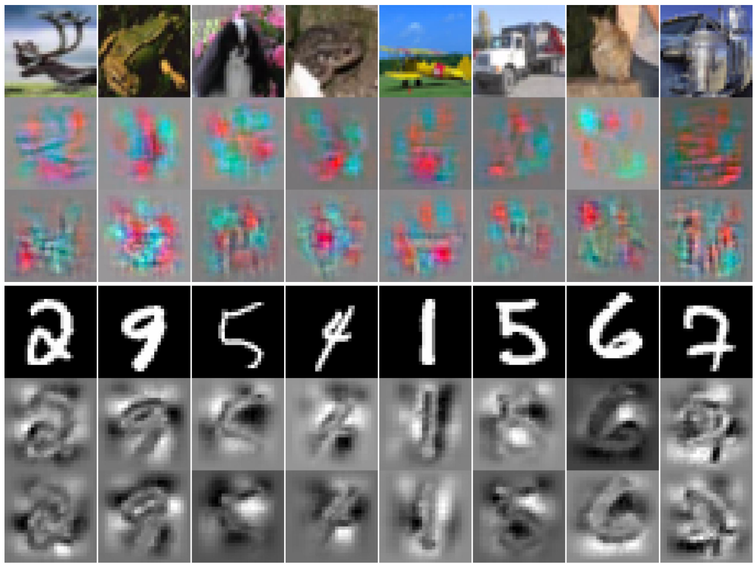


Figure 4.2: Visualizing gradients for an ensemble of k = 2 classifiers trained on CIFAR-10 (top block) and MNIST (bottom block) using GPMR. First, second, and third row in each block illustrates the inputs to the model and gradients of the first and second classifiers, respectively.

4.3.3 Threat Model in Practice

In the previous section, we formalized a geometric framework to analyze the robustness according to the size of the ensemble, $\mathrm{k} = |\mathcal{F}|$, and the extent of the diversity of the gradient directions. This methodology is built upon the optimization problem in Equation 4.3 which assumes that the adversary must fool all classifiers at the input sample. However, it is not practical to reject all samples which do not have the full agreement of the members. In real-world applications, changing the prediction of a subset of the ensemble, $\mathcal{F}' \subset \mathcal{F}$, is enough to alter the prediction of the ensemble. In this case, the lower bound of the robustness, presented in Theorem 4.3.1, reduces based on $\mathrm{k}' = |\mathcal{F}'|$. Previous defenses based on diversifying predictions [107, 109, 110] or gradients [108] do not control the magnitude of the gradients of the members. Thus, the subset required to be fooled in order to fool the ensemble is often smaller than $\lfloor \frac{|\mathcal{F}|}{2} \rfloor + 1$ since the adversary can fool a set of locally weak classifiers, *i.e.*, members with large gradient magnitudes. In the next section, we propose a gradient magnitude equalization loss that alleviate this problem by enforcing: $|\mathcal{F}'| \geq \lfloor \frac{|\mathcal{F}|}{2} \rfloor + 1$.

4.3.4 Joint Gradient Regularization

Here, we present the *joint gradient phase and magnitude regularization (GPMR)* scheme as a theoretically-grounded approach for improving the robustness of the ensemble against bounded alterations of the input domain. joint gradient phase and magnitude regularization (GPMR) maximizes a lower bound to the robustness of the ensemble, according to Theorem 4.3.1, by jointly regularizing the gradient directions and magnitudes. First, we define the gradient diversity promoting loss to increase the angle between the gradients by forcing the cosine similarity of gradients to approach $\frac{-1}{k-1}$ as:

$$\mathcal{L}_{\mathrm{div}} = \frac{2}{k(k-1)} \sum_{0 \leq i < j \leq k-1} \left(\frac{\langle \nabla f^{i}_{c_i}, \nabla f^{j}_{c_j} \rangle}{||\nabla f^{i}_{c_i}||_2 ||\nabla f^{j}_{c_j}||_2} + \frac{1}{k-1} \right)^2, \tag{4.6}$$

where $\frac{2}{k(k-1)}$ normalizes the loss over the number of the pairs in the ensemble. Second, we define the gradient magnitude regularization loss. To focus on regularizing the joint interactions of the members, we opt to equalize the gradient magnitudes rather than minimizing them as:

$$\mathcal{L}_{\mathrm{eq}} = \frac{1}{k} \sum_{i} (||\nabla f^{i}_{c_i}||_2 - \frac{1}{k} \sum_{j} ||\nabla f^{j}_{c_j}||_2)^2. \tag{4.7}$$

This forces the gradient magnitudes to be roughly equal at each input sample and equalizes the contribution of the members to the ensemble robustness. Consequently, the adversary must fool at least the majority of classifiers and cannot fool the ensemble prediction by fooling a few classifiers with the maximum magnitude of gradient at the input sample. The equalization loss also makes GPMR orthogonal to defenses developed for single classifiers controlling the smoothness of the predictions [3, 10, 11, 32, 40, 86, 117, 118, 123]. Hence, other defenses can be employed to further robustify the ensemble by alleviating the vulnerability of individual members. It must be noted that $\mathcal{L}_{\mathrm{eq}}$ does not constrain the magnitude of gradients at two different input samples.

The final loss function for training the ensemble is:

$$\mathcal{L}_{\mathrm{t}} = \mathcal{L}_{\mathrm{xent}} + \lambda_{\mathrm{eq}} \mathcal{L}_{\mathrm{eq}} + \lambda_{\mathrm{div}} \mathcal{L}_{\mathrm{div}}, \tag{4.8}$$

where λ_{div} and λ_{eq} are Lagrangian coefficients controlling the importance of the regularization terms. $\mathcal{L}_{\mathrm{xent}}$ is the classification loss function which computes the average of the cross-entropy loss over all members. The classification loss can be defined on natural or adversarial examples. The latter case combines the proposed framework with adversarial training to further improve the robustness of the ensemble as studied in Section 4.4.3. It may be noted that GPMR does not improve the robustness of the member classifiers. Indeed, it regularizes the interactions of the members to mitigate the adversarial behavior using the joint evaluation of the members. Moreover, GPMR aims to construct an ensemble classifier for which the perturbations crafted for one classifier have less effect on other classifiers or even increase the score corresponding to their predicted class.

Table 4.1: Classification error rate (%) on natural examples. Ensembles consist of k = 3 Conv/ResNet-20 classifiers, *i.e.*Net 1, Net 2, and Net 3. The maximum standard deviation of error is 0.5%, 0.6%, 5.3%, and 0.7% for Base., ADP, GAL, and GPMR, respectively.

Dataset	Classifier	Base.	ADP	GAL	GPMR
MNIST	Net 1	0.76/0.40	0.70/0.41	1.18/0.91	0.72/0.53
	Net 2	0.78/0.43	0.67/0.48	1.14/0.96	0.78/0.57
	Net 3	0.75/0.45	0.69/0.44	1.12/0.93	0.71/0.52
	Ensemble	0.73/0.36	**0.66/0.31**	1.02/0.86	0.71/0.51
CIFAR-10	Net 1	10.43/8.50	10.72/8.93	11.92/9.58	10.37/9.11
	Net 2	10.18/8.15	10.25/9.38	11.58/10.33	10.87/9.52
	Net 3	10.50/8.72	10.80/9.28	11.45/10.19	11.05/9.83
	Ensemble	9.28/6.94	**9.12/6.85**	11.40/9.16	9.30/7.22
CIFAR-100	Net 1	39.83/34.00	43.33/39.35	45.03/39.61	40.46/36.85
	Net 2	38.65/35.58	43.45/40.53	42.32/37.77	40.47/37.48
	Net 3	40.51/35.82	42.94/40.81	43.49/41.13	41.66/37.15
	Ensemble	36.35/30.72	**35.48/30.41**	39.61/36.42	36.76/31.05

4.4 Experiments

Here, we provide the experimental results to evaluate the effectiveness of GPMR. We evaluate the joint robustness of multiple classifiers on the MNIST, CIFAR-10, and CIFAR-100 datasets. We consider two base network architectures detailed in Table 4.4. We train models using stochastic gradient descent with momentum equal to 0.9 and weight decay of $5e-4$. The initial learning rate is set to 10^{-1}, and decayed with the factor of 0.2 every 30 epochs until the final learning rate 10^{-4}. We run the training process for 60 epochs on MNIST, and 200 epochs on CIFAR-10 and CIFAR-100. The batch size for training models is set to 64 for all experiments. We observe that λ_{div} directly affects the classification accuracy of the ensembles as depicted in Figure 4.3a. Consequently, the diversity loss coefficient, λ_{div}, is set to 0.1 for MNIST and 0.04 for CIFAR-10 and CIFAR-100. We also observe that the accuracy of the ensembles on natural examples is roughly independent of λ_{eq}. This is expected since the equalization loss does not minimize the magnitude of gradients. Hence, we select $\lambda_{\text{eq}} = 10$ for all network architectures and datasets based on the experiments conducted in Section 4.4.2 and Figure 4.3d.

We compare our method to three ensemble models. The first ensemble is trained without any diversity encouraging criterion, *i.e.*, GPMR with $\lambda_{\text{eq}} = \lambda_{\text{div}} = 0$. The second ensemble is trained using GAL [108] which diversifies the gradient of predictions. ADP [107] is used as the third method to promote the prediction diversity among classifiers. Due to the constraints on the number of classifiers in ADP, we conduct the comparisons with this baseline on ensembles of $k = 3$ classifiers. For GAL, the coefficient of the diversity loss, λ, is set to 0.5. For ADP coefficients α and β are set to 2 and 0.5, respectively. These values are associated with the best performance reported by the authors. Major parts of our experiments are adapted from Pang et al. [107] to provide consistent evaluations for the future works. The results are the average of 10 independent runs.

4.4.1 Performance on Natural Examples

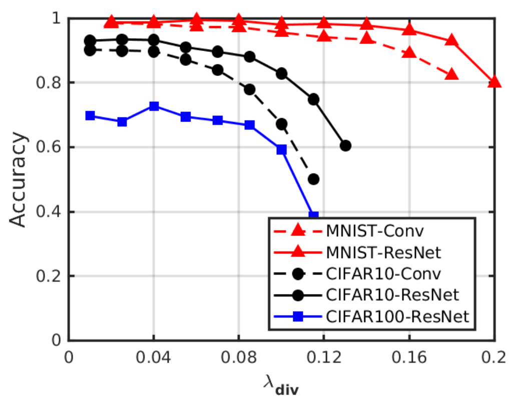
Table 4.1 presents the classification error rate of the member classifiers and the ensembles. Promoting diversity of gradient directions slightly degrades the classification performance

Table 4.2: Classification accuracy (%) for adversarial examples on MNIST. The results for Conv and ResNet architectures are separated by '/'. The coefficient ϵ for JSMA is set to 0.1. The coefficient β of the EAD attack is set to 0.01. The maximum standard deviation of results is 6.3% and 0.20% for GAL and other methods, respectively.

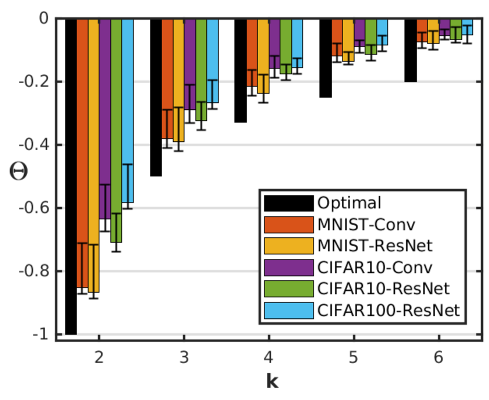
Attack	Setting	**MNIST**				
		Baseline	ADP$^{k=3}$	GAL$^{k=3}$	GPMR$^{k=2}$	GPMR$^{k=3}$
FGSM	$\epsilon=0.1$	65.9/75.2	83.5/95.2	57.4/84.3	85.0/92.2	**90.8/97.6**
	$\epsilon=0.2$	18.2/20.6	45.1/51.2	31.9/39.2	38.9/54.1	**58.7/65.4**
BIM	$\epsilon=0.1$	46.5/50.0	72.5/88.9	40.0/59.2	75.1/80.9	**89.0/92.4**
	$\epsilon=0.15$	12.1/13.8	68.0/72.7	41.5/51.3	67.0/74.3	**73.8/79.6**
PGD	$\epsilon=0.1$	48.6/49.4	78.7/82.4	53.8/54.6	72.8/77.0	**84.8/87.6**
	$\epsilon=0.15$	4.3/7.6	38.2/41.1	26.7/30.8	36.9/42.5	**51.4/59.3**
MIM	$\epsilon=0.1$	54.1/57.6	88.7/91.5	75.0/84.2	90.8/92.1	**91.3/93.5**
	$\epsilon=0.15$	6.4/15.9	70.9/76.8	64.4/69.6	74.1/79.8	**76.5/82.4**
JSMA	$\gamma=0.3$	79.5/83.1	90.1/95.0	76.4/83.6	90.7/94.0	**95.0/96.7**
	$\gamma=0.6$	73.2/75.0	86.2/89.8	72.4/81.3	85.9/87.6	**92.8/93.3**
C&W	$c=1.0$	25.4/31.3	73.2/78.5	52.7/55.2	77.0/79.4	**80.4/82.4**
	$c=10.0$	4.6/5.8	20.1/24.0	10.9/15.7	22.3/27.8	**28.5/33.4**
EAD	$c=5.0$	25.1/28.4	90.2/93.0	72.4/74.4	90.2/90.5	**92.8/96.1**
	$c=10.0$	7.1/7.4	86.6/89.6	68.9/72.3	83.2/85.1	**87.6/91.9**

Table 4.3: Classification accuracy (%) for adversarial examples on CIFAR-10. The results for Conv and ResNet architectures are separated by '/'. The coefficient ε for JSMA is set to 0.2 for CIFAR-10, respectively. The coefficient β of the EAD attack is set to 0.01. The maximum standard deviation of results is 6.3% and 0.20% for GAL and other methods, respectively.

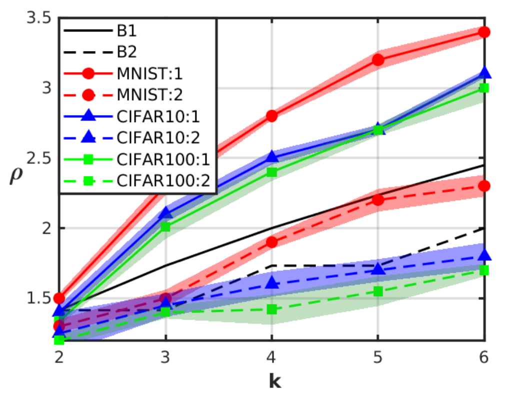
Attack	Setting	CIFAR-10				
		Baseline	ADP$^{k=3}$	GAL$^{k=3}$	GPMR$^{k=2}$	GPMR$^{k=3}$
FGSM	$\varepsilon=0.02$	30.3/35.2	50.5/60.4	27.6/34.9	53.2/55.2	**61.0/66.8**
	$\varepsilon=0.04$	17.6/18.0	44.0/48.7	31.3/32.9	43.1/45.8	**56.0/60.5**
BIM	$\varepsilon=0.01$	15.4/16.9	41.8/43.9	31.8/33.8	45.5/48.5	**50.3/55.2**
	$\varepsilon=0.02$	5.7/7.2	23.6/32.5	20.4/23.9	33.1/34.7	**38.6/46.2**
PGD	$\varepsilon=0.01$	16.9/23.1	44.4/49.2	27.4/36.9	44.8/53.8	**62.5/64.9**
	$\varepsilon=0.02$	6.5/7.5	23.1/31.6	20.4/21.5	24.0/33.9	**35.8/49.2**
MIM	$\varepsilon=0.01$	22.3/24.2	46.3/54.6	43.3/47.6	49.0/58.4	**63.4/66.8**
	$\varepsilon=0.02$	6.8/7.4	25.0/33.7	21.2/28.9	29.3/35.5	**47.2/51.9**
JSMA	$\gamma=0.05$	25.9/29.0	40.1/43.7	35.7/36.7	43.3/45.9	**52.9/55.4**
	$\gamma=0.1$	23.9/26.2	31.2/38.2	32.8/35.7	35.6/40.2	**48.1/50.6**
C&W	c=0.01	41.8/46.3	50.9/54.8	32.3/36.6	53.5/58.0	**60.9/66.9**
	c=0.1	15.8/18.5	22.6/25.4	18.7/20.4	22.1/27.3	**32.9/35.1**
EAD	c=1.0	12.3/17.1	65.6/70.4	52.6/54.2	63.0/68.8	**76.6/79.8**
	c=5.0	2.4/3.3	30.1/30.3	10.5/18.5	27.4/31.2	**45.8/50.2**



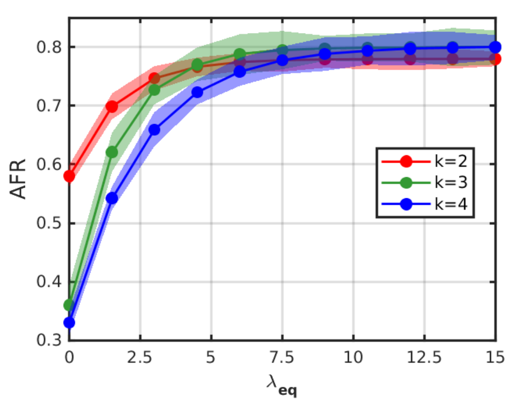
(a)



(b)



(c)



(d)

Figure 4.3: (a) Classification accuracy of ensembles versus λ_{div} on natural examples, (b) expected cosine similarity of gradients versus the number of classifiers, (c) robustness of the ensemble versus the number of classifiers, where B1 and B2 denote the optimal and practical robustness at $\phi = \frac{\pi}{2}$, and the solid and dashed plots show the results for GPMR with λ_{eq} equal to 10 and 0, respectively, (d) average fooling rate of members in the ensemble versus λ_{eq}.

Table 4.4: Network architecture of the base classifiers, consisting of Convolution (C), Max-pooling (M) and Fully-connected (F) layers. Each RES block consists of two (C) with a skip connection. All layers, except the last (F), are followed by ReLU. The number of classes is denoted by m.

Model	Structure
Conv	2×C64-M-2×C128-M-2×C256-M-2×C256-F512-F(m)
ResNet-20	C16-3×RES16-3×RES32-3×RES64-F512-F(m)

on natural examples. This is attributed to that by diversifying gradients classifiers learn to discriminate input samples based on distinct sets of representative features, illustrated in Figure 4.2. Minimizing the similarity of salient regions using the gradient diversity loss divides important features between classifiers which reduces the accuracy on natural examples. However, this enhances the robustness against adversarial examples as presented in experiments on white-box defense performance. Table 4.1 also highlights the superior performance of GPMR compared to GAL. As discussed in Section 4.3.4, GAL does not consider the optimal bound for the similarity of gradients. During the training, it forces the cosine similarity of gradients for k classifiers to approach –1 while the optimal bound is $\frac{-1}{k-1}$. Consequently, GAL suffers from the fluctuation in the loss and accuracy of individual classifiers during the training.

4.4.2 Theory vs. Practice

Here, we evaluate the gap between the theory and practice of the proposed approach. To this aim, we first measure the diversity of the gradient directions within the trained ensemble using the expected cosine similarity as:

$$\Theta(\mathcal{F}) = \frac{2}{k(k-1)}\mathbb{E}_x\Big[\sum_{0\leq i<j\leq k-1} \frac{\langle \nabla f^i_{c_i}, \nabla f^j_{c_j}\rangle}{||\nabla f^i_{c_i}||\cdot||\nabla f^j_{c_j}||}\Big]. \tag{4.9}$$

Figure 4.3b presents the empirical values of the cosine similarity computed over 1,000 test samples. In all experiments, the cosine similarity is negative and close to the optimal value which implies that the diversity of the gradient directions is better than the orthogonal case

Table 4.5: Classification accuracy (%) on adversarial examples for CIFAR-100 with ResNet-20 architecture .

Attack	ε	Base.	ADP$^{k=3}$	GAL$^{k=3}$	GPMR$^{k=2}$	GPMR$^{k=3}$
BIM	0.005	23.6	27.3	21.8	34.2	**37.8**
	0.01	11.7	13.6	12.8	19.5	**24.2**
PGD	0.005	25.2	32.4	30.2	36.1	**38.5**
	0.01	11.4	17.8	14.0	25.5	**29.2**
MIM	0.005	23.4	31.2	26.4	32.8	**37.1**
	0.01	10.3	18.9	16.7	22.5	**28.6**

Table 4.6: Classification accuracy (%) of combined defenses on ResNet-20. The maximum standard deviation is 1.4%.

Defense	FGSM	BIM	PGD	MIM
Def$_A$	41.7	19.6	25.6	28.5
Def$_A$ + GPMR	**70.9**	**54.0**	**55.1**	**58.9**
Def$_B$	41.3	25.4	32.1	33.8
Def$_B$ + GPMR	**66.2**	**68.5**	**57.7**	**62.3**

where all gradients are perpendicular. Diversifying the gradients on MNIST achieves closer values to the optimal bound compared to CIFAR-10 and CIFAR-100. We attribute this to the capacity of members compared to the complexity of the task. Increasing the size of the ensemble enlarges the gap between the practice and the optimal bound of the gradient diversity.

For the second evaluation, we adapt the robustness measure proposed by Moosavi-Dezfooli et al. [3] for the ensemble framework as:

$$\rho(\mathcal{F}) := \mathbb{E}_{\mathrm{x}}\Big[\frac{\Delta(\mathrm{x};\mathcal{F})}{\max_{\mathrm{f}\in\mathcal{F}}\Delta(\mathrm{x};\mathrm{f})}\Big], \tag{4.10}$$

where $\Delta(\mathrm{x};\mathcal{F})$ is the minimum ℓ_2-norm adversarial perturbation for the given classifier $\mathcal{F}$ at x, and we approximate it using ℓ_2-DeepFool [3]. Indeed, $\rho(\mathcal{F})$ measures the expected ratio of the robustness of the ensemble over the robustness of the most robust classifier in the set. This measure can reliably characterize the effectiveness of a defense based on ensembles since it measures the relative robustness of the set compared to its members. We compute this measure over $1,000$ test examples. Figure 4.3c illustrates the results for this evaluation. GPMR improves the robustness on all datasets as the size of the ensemble grows. For instance, with $\mathrm{k} = 4$ classifiers, GPMR increases the magnitude of the minimum ℓ_2 perturbation by **2.75**, **2.5**, and **2.4** on MNIST, CIFAR-10, and CIFAR-100, respectively. We also ablate the role of gradient magnitude equalization by repeating this evaluation using GPMR with $\lambda_{\mathrm{eq}} = 0$. As depicted in Figure 4.3c, diversifying the gradients without equalizing the gradient magnitudes significantly limits the effectiveness of GPMR.

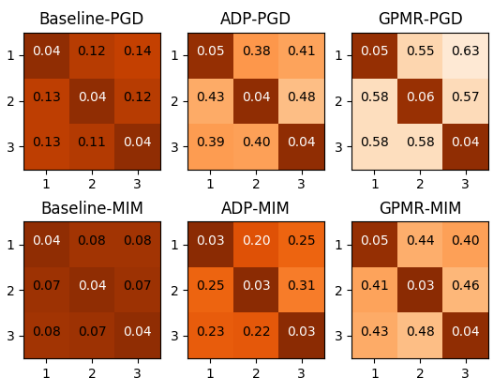
We further analyze the role of the gradient equalization loss by measuring the average ratio of the number of classifiers that are fooled by DeepFool [3] over the number of members in the ensemble. We refer to this ratio as the average fooling ratio (AFR) of the members. We train ensembles consist of $\mathrm{k} = \{2, 3, 4\}$ ResNet-20 networks on CIFAR-10. Figure 4.3d presents the results for these experiments. We observe that without the equalization loss ($\lambda_{\mathrm{eq}} = 0$) AFR is 0.58, 0.37, and 0.34 for k equal to 2, 3, and 4, respectively. This illustrates that the attack targets merely 1 or 2 classifiers at each input sample to fool the ensemble prediction. However, by increasing λ_{eq} AFR improves significantly, which validates the effectiveness of the gradient equalization loss for regularizing the contribution of members.

4.4.3 White-box Defense Performance

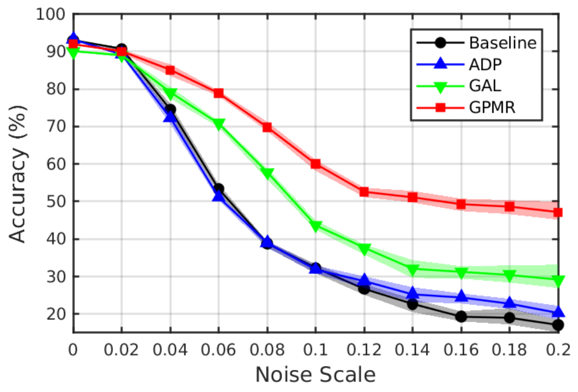
We evaluate the performance of GPMR against several well-known and powerful white-box attacks including fast gradient sign method (FGSM) [10], basic iterative method (BIM) [11], projected gradient descent (PGD) [86], momentum iterative method (MIM) [56], Jacobian-based saliency map attack (JSMA) [57], Carlini & Wagner (C&W) [58], and elastic-net attack (EAD) [124]. A brief summary of these attacks can be found in [107]. For each attack, as detailed in Tables 4.2,4.3, and 4.5, we consider two settings to demonstrate the effectiveness of our approach against a wide range of adversaries. For BIM, PGD, and MIM, the iteration of attack is set to 10 and the step size is set to $\frac{\varepsilon}{10}$. Both C&W and EAD are implemented with the learning rate of 0.01 and $1,000$ iterations.

Tables 4.2, 4.3, and 4.5 present the classification accuracy of ensemble models on adversarial examples. GPMR consistently outperforms other ensemble-based defenses on both network architectures and all datasets. Ensembles consist of $\mathrm{k} = 2$ classifiers trained with GPMR outperform GAL ensembles with $\mathrm{k} = 3$ classifiers, and provide comparable performance to ADP ensembles with $\mathrm{k} = 3$ classifiers on MNIST and CIFAR-10. On CIFAR-100, our ensemble model with $\mathrm{k} = 2$ classifiers surpasses all other ensembles consisted of $\mathrm{k} = 3$ classifiers. This can better demystify the effectiveness of GPMR since its functionality is independent of the number of classes in the task. However, the number of classifiers required by ADP increases as the number of classes grows.

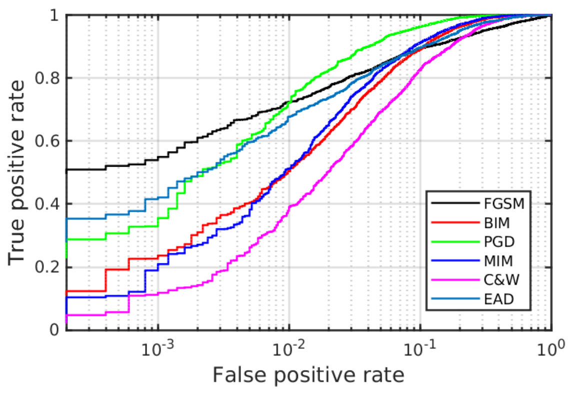
In another set of experiments, we evaluate the orthogonality of GPMR to other defenses. We consider adversarial training on FGSM (Def_A) [10] and PGD (Def_B) [86] to combine with GPMR. Both defenses are implemented using the same training setup as GPMR. The ℓ_∞-norm magnitude of perturbations, ε, is uniformly sampled from the interval $[0.01, 0.05]$ as suggested by Kurakin et al. [125]. Table 4.6 shows the classification accuracy of combined defenses against FGSM ($\varepsilon = 0.04$), BIM ($\varepsilon = 0.02$), PGD ($\varepsilon = 0.02$), and MIM ($\varepsilon = 0.02$) on CIFAR-10. As we observe, the combination of other defenses with GPMR consistently improves the performance of the defense. This is attributed to GPMR equalizing gradients and not reducing their magnitudes, while the conventional defense methods seek to reduce the magnitude of gradients. Hence, they can be combined to simultaneously diversify gradient



(a)



(b)



(c)

Figure 4.4: (a) Transferability of adversarial examples in ensembles of size k = 3 on CIFAR-10. The rows and columns illustrate the source and target networks, respectively, (b) Gradient-free evaluation of robustness using Gaussian random noise, and (c) ROC curves for detecting adversarial examples using the standard deviation of predictions.

Table 4.7: Detection performance of ensembles on CIFAR-10 using AUC (10^{-2}) score. Results for ADP are cited from the original paper.

Attack	Setting	ADP	GAL	GPMR
FGSM	$\varepsilon = 0.1$	91.19	90.98	**95.29**
BIM	$\varepsilon = 0.1$	93.14	90.54	**96.32**
PGD	$\varepsilon = 0.1$	97.03	93.15	**98.45**
MIM	$\varepsilon = 0.1$	94.09	91.24	**94.13**
C&W	$c = 1.0$	90.98	88.46	**93.67**
EAD	$c = 20.0$	94.84	91.52	**96.46**

directions, equalize gradient magnitudes, and reduce gradient magnitude. Combining GPMR with adversarial training further improves the robustness since the lower bound in Theorem 4.3.1 improves when the magnitude of gradients decreases.

4.4.4 Transferability Across Individual Classifiers

Defensive interactions between several classifiers can be characterized by the transferability of adversarial examples among them. We perform transferability experiments using PGD and MIM which are powerful attacks for the black-box setting [107,126]. We compute adversarial examples for each member classifier and then evaluate their transferability across other members by computing the classification accuracy of the target classifier. The perturbation size for both attacks is set to $\varepsilon = 0.05$. Figure 4.4a presents the results for ResNet-20 architecture on CIFAR-10 and suggests that diversifying gradients is an effective approach to reduce the transferability of adversarial examples among members in the ensemble. However, it may be noted that minimizing the transferability among every two members does not lead to the maximum robustness of the ensemble since for $k > 2$ the optimal cosine similarity of pair of gradients is greater than -1.

4.4.5 Gradient-free Evaluation of the Robustness

White-box attacks are not sufficient to assess the performance of a defense method since the defense may cause obfuscated gradients and mislead the evaluation [127]. Therefore, in Figure 4.4b, we evaluate the performance of ensembles consist of $k = 3$ ResNet-20 classifiers on CIFAR-10 samples augmented with random noise. The maximum standard deviation of the results is 2.1%, 2.3%, 4.5%, and 2.3% for baseline, ADP, GAL, and GPMR, respectively. Results suggest that diversifying the predictions in ensembles does not improve the robustness to random perturbations since the performance of ADP is similar to the baseline. However, diversifying gradients improves the robustness to noisy input samples which demonstrates the superiority of gradient diversity compared to prediction diversity.

4.4.6 Joint Robustness for Detecting Adversaries

Here, we adopt a measure based on the prediction of all members to evaluate the detection performance of ensembles trained by GPMR. We compute the standard deviation of the class probability scores associated with the predicted class over all the classifiers and compare it with a predefined threshold to accept or reject the input example. Figure 4.4c and Table 4.7 present the receiver operating characteristic (ROC) curves and AUC scores for the detection performance on $1,000$ natural examples and $1,000$ adversarial examples from the CIFAR-10 dataset. All ensembles consist of $\mathrm{k} = 3$ classifiers. Results validate the performance of our model on detecting alterations of the input samples. Moreover, ADP outperforms GAL due to the disparity in the robustness of subsets of classifiers in GAL. We observe that GAL causes a notable robustness gap between the most and least robust sets of classifiers in the ensemble since it does not regularize the contribution of members in the ensemble.

4.5 Conclusion and Future work

In this paper, we introduced a practically feasible scenario of first-order defensive interactions between members of an ensemble. We both theoretically and empirically demonstrated that imposing these interactions significantly improves the robustness of ensembles. We proposed the joint gradient phase and magnitude regularization (GPMR) as an empirical tool to regularize the interaction between members and equalize their role in the ensemble decision. Furthermore, we concluded that the superior performance of GPMR is due to its capability to increase the effective number of members contributing to the robustness. For the future work, we plan to analyze the gradient diversity loss in more detail. We mainly aim to address the deterioration of the classification accuracy on the natural examples. Hence the effect of several factors such as the model capacity and the intrinsic characteristics of the gradients in deep models will be explored. Moreover, employing multiple classifiers to improve the robustness of the classification can cause notable computational costs. Hence, we also plan to evaluate the possible choices of the network architectures that can reduce the capacity of the final ensemble while preserve the flexibility of the model for promoting the gradient diversity.

Chapter 5

Revisiting Outer Optimization in Adversarial Training

5.1 Introduction

Susceptibility of DNNs to manipulated inputs has raised critical concerns regarding their deployment in security-sensitive applications [11,128,129]. The worst-case manipulation can be characterized by *adversarial examples*: carefully crafted input examples that can easily alter the model prediction while remaining benign to the human perception [32,42]. A principal approach to formalize the imperceptibility is to bound the perturbation using ℓ_{p}-norm. Hence, the problem of finding a model robust to adversarial manipulation reduces to finding the one that generalizes well merely on the bounded neighborhood of the input example. Although this task seems effortless for humans, achieving such invariance is notoriously difficult for DNNs. The reason for this behavior has not fully understood yet, but several factors have shown to be influential, including the high cardinality of the data space [105,106].

One of the most effective methods (defenses) to alleviate adversarial susceptibility is AT which improves the robustness by training the model on the worst-case loss [42,43]. Given the deep model $\mathrm{F}_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ and the surrogate loss function for the empirical adversarial risk L, the training objective of AT is defined as:

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},\mathrm{y})\sim\mathcal{D}}\left[\mathrm{L}^*\Big(\mathbf{x},\mathrm{y};\boldsymbol{\theta}\Big)\right], \tag{5.1a}$$

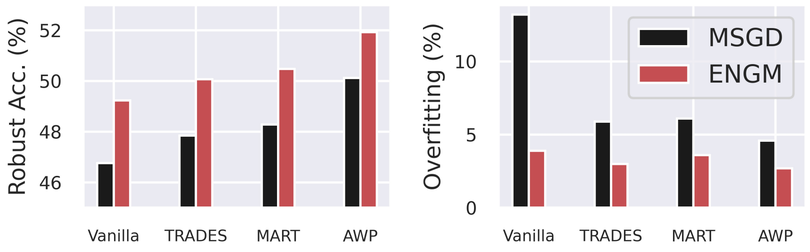


Figure 5.1: Replacing MSGD with ENGM for outer optimization in AT results in consistent improvement of robust accuracy and generalization.

$$\mathrm{L}^*\Big(\mathbf{x}, \mathrm{y}; \boldsymbol{\theta}\Big) = \max_{||\mathbf{x}-\mathbf{x}'||_p \leq \varepsilon} \mathrm{L}\Big(\mathrm{F}_{\boldsymbol{\theta}}(\mathbf{x}'), \mathrm{y}\Big), \tag{5.1b}$$

where the input example $\mathbf{x}$ and the corresponding label y are a sample from the data distribution $\mathcal{D}$, $\mathbf{x}'$ is the adversarial equivalent of $\mathbf{x}$, and ε is the maximum ℓ_p-norm magnitude of the perturbation. Throughout the paper, we refer to Equations 5.1a and 5.1b as the *outer* and *inner* optimization in AT, respectively. The inner optimization finds the worst-case adversarial example and the outer optimization minimizes the empirical adversarial risk over the network parameters, $\boldsymbol{\theta}$. AT is known to be intrinsically more challenging than natural training (NT), e.g. adversarially robust models require enormous over-parameterization [43, 130].

Numerous efforts have been devoted to the development of AT analyzing its different aspects, such as the inner optimization [43, 131, 132], adversarial objective [133–135], computational cost [136–138], and evaluation methods [3, 13, 56, 58, 127]. Recent studies on the topic have revealed two major shortcomings of AT which contradicts common observations on NT. First, AT severely induces overfitting [139, 140], referred to as *robust overfitting*, whereas in NT overfitting is known to be less prominent especially in over-parameterized models [141–143]. Second, AT is highly sensitive to hyperparameter setting, e.g. a slight change in the weight decay can drastically change the robust performance [144, 145].

The majority of the previous works on AT have analyzed the inner optimization and its properties. However, the potential impact of outer optimization on the performance and shortcomings of AT has been critically overlooked. Furthermore, the success of the two recent SOTA approaches of AT which indirectly affect the outer optimization by weight perturbations [146] or weight smoothing [140] advocates for further investigation on outer optimization. Based on these observations, we raise a fundamental question regarding outer

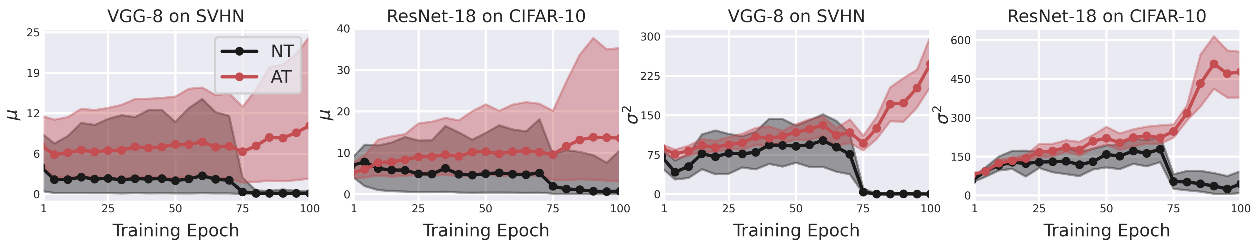


Figure 5.2: Expected norm (μ) and variance (σ^2) of gradients during NT and AT. Learning rate is decayed from 10^{-1} to 10^{-2} at epoch 75. Note that the norm and variance in AT is higher than NT and escalates after learning rate decay.

optimization in AT and attempt to address it in this work:

Is the conventional MSGD, developed for non-convex optimization in NT, a proper choice for the outer optimization in AT? **If not**, *what modifications are required to make it suitable for the AT setup?*

To answer the first question, we empirically evaluate and compare two statistical parameters of gradients, namely expected norm and expected variance, in NT and AT. Both these parameters are known to be major determinants of the performance of MSGD in NT [147–149]. We find that they are notably higher in AT compared to NT. Furthermore, after decaying the learning rate in NT, both the gradient norm and variance deteriorate suggesting convergence to a local minimum. However, in AT, they escalate after the learning rate decay. These observations highlight substantial disparities between the characteristics of the gradients in AT and NT. Consequently, we argue that MSGD, developed essentially for NT, is not the most proper choice for outer optimization in AT since it is not designed to be robust against high gradient norm and variance.

Motivated by these observations, the current work attempts to develop an optimization method that is more suitable for AT, *i.e.*less sensitive to the gradient norm and variance. The contributions of the paper are as follows:

- We investigate the effect of AT on gradient properties and provide empirical evidence that AT induces higher gradient norm and variance. We argue that this hinders the optimization since the convergence rate of MSGD is highly dependent on the variance of the gradients.

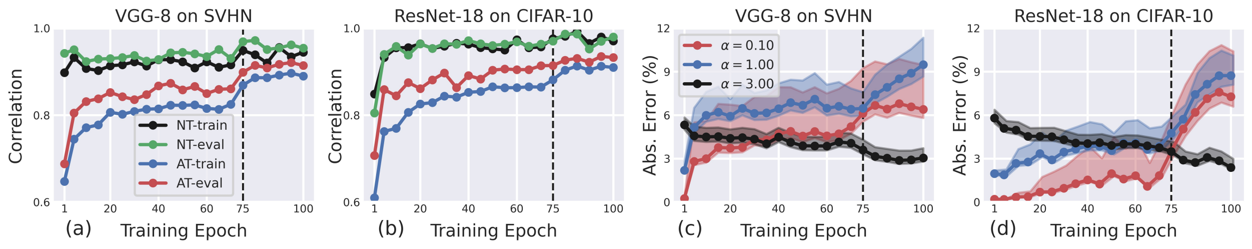


Figure 5.3: (a,b): Characterizing the linear correlation between $||\nabla_{x_i} L_i||$ and $||\nabla_\theta L_i||$ using Pearson correlation coefficient. (c, d): The absolute value of error (%) for estimating w_i using Equation 5.7. Dashed black line denotes the learning rate decay from 10^{-1} to 10^{-2}.

- We propose an optimization method tailored specifically for AT, termed ENGM, whose convergence rate is independent of the gradient variance.
- We empirically analyze the norm of gradients and provide insightful observations regarding their correlation in DNNs. Harnessing this, we develop a fast approximation to ENGM that significantly alleviates its computational complexity.
- Through extensive evaluations and ablation studies, we demonstrate that the proposed optimization technique consistently improves the performance and generalization of the SOTA AT methods.

5.2 Analyzing Outer Optimization in AT

We first describe the notations in Section 5.2.1. Then, we analyze the properties of gradients in AT and NT in Section 5.2.2. We briefly review MSGD and one of its variants that is less sensitive to adversarial gradients in Section 5.2.3. In Section 5.2.4, we describe our proposed optimization technique whose convergence rate is independent of the variance of the gradients. Later in Section 5.2.5, we reveal an interesting phenomenon in DNNs that enables us to approximate a fast version of the proposed optimization technique.

5.2.1 Notations

Throughout the paper, we denote scalars, vectors, functions, and sets using lower case, lower case bold face, upper case, and upper case calligraphic symbols, respectively. We use notation $||\cdot||_p$ for the ℓ_p-norm and drop the subscript for $p = 2$. We employ the commonly used cross-entropy loss as the empirical risk and denote the loss on i^{th} example, $L(F_\theta(\mathbf{x}_i), y_i)$, as L_i for the sake of brevity.

5.2.2 Comparison of Gradient Properties

We experiment to analyze two statistical parameters of gradients which are major determinants in the performance of MSGD in NT and compare them with those in the setup of AT. The first parameter is the expected norm of gradients $\mu := \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[||\nabla_\theta L\Big(F_\theta(\hat{\mathbf{x}}), y\Big)||\Big]$, where $\hat{\mathbf{x}}$ is the natural example, $\mathbf{x}$, in NT and the adversarial example, $\mathbf{x}'$, in AT. Change in the expected norm of gradients directly affects the learning rate, the most important hyper-parameter in NT [147, 148]. The second parameter is the upper bound for the variance of gradients, and is defined as:

$$\sigma^2 := \sup_\theta \ \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\Big|\Big|\nabla_\theta L\Big(F_\theta(\hat{\mathbf{x}}), y\Big) - \bar{\mathbf{g}}\Big|\Big|^2\Big], \tag{5.2}$$

where $\bar{\mathbf{g}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\nabla_\theta L\Big(F_\theta(\hat{\mathbf{x}}), y\Big)\Big]$. It is shown that the convergence of MSGD is $O(\sigma^2)$ [150]. We roughly estimate both parameters during the training by averaging the gradient properties over $1,000$ training examples at the end of training epochs. NT and AT are performed with ResNet-18 and VGG-8 on CIFAR-10 and SVHN datasets, respectively. Inner optimization in AT follows the standard setup, *i.e.*10 steps of ℓ_∞-norm PGD with $\epsilon = 8/255$ and step size $\epsilon/4$ to maximize the adversarial loss.

Figure 5.2 demonstrates μ and σ^2 during 100 training epochs with the learning rate decay from 10^{-1} to 10^{-2} at epoch 75. We observe that the expected norm and variance of gradients is notably higher in AT compared to NT. The expected norm also increases continuously in AT which is contrary to NT. After the decay of the learning rate, both parameters decreases significantly in NT suggesting the convergence of the training to a local minima. However in AT, the expected norm grows and the variance increases drastically. These findings highlight substantial disparities between the characteristics of the gradients in AT and NT. In the next section, we theoretically analyze how these differences can affect the convergence of MSGD.

5.2.3 Revisiting Stochastic Gradient Descent

In this part, we analyze the functionally and convergence of MSGD to identify modifications that improves its suitability for the AT setup. The update rule of MSGD at iteration t is as follows:

$$\mathbf{v}_{t+1} = \beta \mathbf{v}_t + \frac{1}{|\mathcal{I}_t|} \sum_{i \in \mathcal{I}_t} \nabla_\theta L_i, \tag{5.3a}$$

$$\theta_{t+1} = \theta_t - \eta \mathbf{v}_{t+1}, \tag{5.3b}$$

where η is the learning rate, $\mathbf{v}_{t+1}$ is the Polyak's momentum with the corresponding modulus β [151], $\mathcal{I}_t$ is the randomly selected set of indices for the mini-batch with size $|\mathcal{I}_t|$, and L_i is the objective for optimization computed on the i^{th} example. Assuming F has bounded variance of gradients according to Equation 5.2, and is smooth in θ, *i.e.*$F_{\theta_1}(\mathbf{x}) \leq F_{\theta_2}(\mathbf{x}) + \langle \nabla_\theta F_{\theta_1}(\mathbf{x}), \theta_2 - \theta_1 \rangle + \frac{c}{2}||\theta_2 - \theta_1||^2$, Yu et al. [150, 152] have shown that the convergence rate of MSGD for non-convex optimization in DNNs is $O(\sigma^2)$. Hence, MSGD is not suitable for tasks with high gradient variance. Intuitively, higher variance implies that the gradients are not aligned with the average gradients which are being used to update the model parameters. This hinders the optimization process since the update is merely favorable for a portion of examples in the mini-batch.

One alternative to MSGD that is less sensitive to the variance of the gradients is normalized gradient descent with momentum (SNGM) [149]. SNGM is shown to provide better generalization for training with large batch size, *i.e.*another cause of high gradient variance. Concretely, SNGM modifies Equation 5.3a as:

$$\mathbf{v}_{t+1} = \beta \mathbf{v}_t + \frac{\sum_{i \in \mathcal{I}_t} \nabla_\theta L_i}{|| \sum_{i \in \mathcal{I}_t} \nabla_\theta L_i ||}, \tag{5.4}$$

which limits the gradient norm by normalizing the magnitude of mini-batch gradients and considers only the direction of the average gradient. Zhao et al. [149] have shown that the convergence of SNGM is $O(\sigma)$, and therefore, is more suitable for tasks with induced gradient fluctuations. We also observe in Section 5.3.1 that SNGM improves the generalization in

Algorithm 3 Fast ENGM

```
1: Initialize $\tau > 0$, $\beta_\gamma \in [0, 1)$, $\alpha > 0$, $\gamma_0 = 0$, $\gamma_1 = 1$, Boolean parameter *Naive*.
2: for $t = 0 \dots t_1 - 1$ do
3:     Compute $L_i$, $\forall i \in \mathcal{I}_t$;                                    ▷ inner optimization
4:     Compute $\mathcal{G}_{x,t} = \{\nabla_x L_i : i \in \mathcal{I}_t\}$;                               ▷ backprop. ×1
5:     if mode$(t, \tau) = 0$ and Naive = False then
6:         Compute $\mathcal{G}_{\theta,t} = \{\nabla_\theta L_i : i \in \mathcal{I}_t\}$;       ▷ backprop. ×n every $\tau$ iterations
7:         $\gamma_1', \gamma_0' = \text{LinearRegression}(\mathcal{G}_{x,t}, \mathcal{G}_{\theta,t})$          ▷ estimate slope and intercept
8:         $\gamma_0 \leftarrow \beta_\gamma \gamma_0 + (1 - \beta_\gamma)\gamma_0'$, and $\gamma_1 \leftarrow \beta_\gamma \gamma_1 + (1 - \beta_\gamma)\gamma_1'$;
9:     end if
10:    $\hat{w}_i \leftarrow \max(\frac{\alpha}{||\gamma_1 \nabla_x L_i + \gamma_0||}, 1)$, $\forall i \in \mathcal{I}_t$;
11:    Update $\theta$ with MSGD on the reweighted loss $\frac{1}{|\mathcal{I}_t|}\sum_{i \in \mathcal{I}_t} \hat{w}_i L_i$         ▷ backpropagation×1
12: end for
```

AT. This suggests that reducing the sensitivity of the optimizer to the gradient variance has a direct impact on the generalization and performance of the task with adversarial gradients.

5.2.4 Example-normalized Gradient Descent with Momentum

Although SNGM is less sensitive than MSGD to the variance of gradients, it does not impose any constraint on the variance. Hence, the variance can still become large and impede the optimization. To address this, we introduce a transformation on gradient vectors that bounds the variance of the gradients in the mini-batch and makes the convergence rate of the optimizer independent of the variance.

Theorem 5.2.1. *For any arbitrary distribution $\mathcal{P}$ of random vectors, applying the transformation $T(\mathbf{a}) = \min(\frac{\alpha}{||\mathbf{a}||}, 1)\mathbf{a}$ with $\alpha > 0$ bounds the variance of vectors to $4\alpha^2$.*
(Proof is provided in Section 1 of Supp. material.)

We use the transformation in Theorem 5.2.1 to bound the variance of the gradients. To this aim, we rewrite Equation 5.3a as:

$$\mathbf{v}_{t+1} = \beta \mathbf{v}_t + \frac{1}{|\mathcal{I}_t|}\sum_{i \in \mathcal{I}_t} w_i \nabla_\theta L_i, \tag{5.5a}$$

$$w_i = \min\Big(\frac{\alpha}{||\nabla_\theta L_i||}, 1\Big), \tag{5.5b}$$

Method	MSGD	MSGD+GNC	SNGM	F-ENGM	N-ENGM	A-ENGM	ENGM
Ex. time (sec./iter)	0.60	0.61	0.63	5.05	0.75	0.83	5.06

Table 5.1: Execution time of the outer optimization methods. Experiments are conducted on an NVIDIA Titan-RTX GPU.

where w_i is the normalizing coefficient for $\nabla_\theta L_i$, and α is the maximum allowed norm of gradients. This update rule limits the maximum norm of the gradients on each input example to α. Hence, it prevents high magnitude gradients from dominating the updating direction and magnitude in the mini-batch. It might be noted that α scales with the square root of the model size, and larger models require higher values of α. We refer to this approach as **e**xample-**n**ormalized stochastic **g**radient descent with **m**omentum (ENGM). ENGM recovers MSGD when $\alpha \gg 1$. The convergence properties of ENGM is analyzed in Theorem 5.2.2.

Theorem 5.2.2. *Let* $A(\theta)$ *be the average loss over all examples in the dataset, and assume that it is smooth in* θ*. For any* $\alpha > 0$ *and total iterations of* t_1*, optimizing* $A(\theta)$ *using ENGM (Equation 5.5) has the convergence of* $O(\alpha)$*. (Proof is provided in Section 1 of Supp. material.)*

Theorem 5.2.2 shows that the convergence rate of ENGM is $O(\alpha)$ and is independent of the variance of gradients. Hence, it is suitable for optimizing objectives with high gradient variance. Later in Section 5.3.1, we empirically validate this and show that the enhanced regularization of ENGM provides better optimization compared to SNGM and MSGD for AT. Despite the intrinsic merits of ENGM, it is computationally expensive since evaluating each w_i requires a dedicated backpropagation and cannot be implemented in parallel. In particular, Equation 5.5 requires $|\mathcal{I}_t|$ backpropagation for each mini-batch. In the next section, we discuss an empirical observation on the gradients of DNNs that enables us to estimate w_i and consequently Equation 5.5 using merely one additional backpropagation.

5.2.5 Accelerating ENGM via Gradient Norm Approximation

During our evaluations, we observe an interesting phenomenon that enables us to develop a fast approximation to ENGM. Particularly, we observe that the norm of gradients w.r.t.

the network parameters, $||\nabla_\theta L_i||$, is linearly correlated with the norm of the gradients w.r.t. the input example, $||\nabla_{x_i} L_i||$. To illustrate this phenomenon, we track both gradient norms on $1,000$ training examples during NT and AT using VGG-8 on SVHN and ResNet-18 on CIFAR-10. We compute Pearson correlation coefficient to measure the correlation between the two norms. Figures 5.3a and 5.3b show the correlation coefficient during AT and NT with the model in the evaluation and training modes. We can see that there is a significant correlation between the two norms in DNNs which becomes stronger as the training proceeds. The correlation exists in both the training and evaluation modes of the model, and is slightly affected by the update in the statistics of the batch normalization modules.

Harnessing this phenomenon, we can estimate the norm of gradient w.r.t. the network parameters (computationally expensive) using the norm of gradients w.r.t. the inputs (computationally cheap) with a linear approximation as:

$$||\nabla_\theta L_i|| \approx \gamma_1 ||\nabla_{x_i} L_i|| + \gamma_0, \tag{5.6}$$

where γ_0 and γ_1 are coefficients for the slope and intercept of the linear estimation, respectively. Employing this estimation, we can approximate the functionality of ENGM by a simple modification of the loss on the ith input example, L_i, and keeping the popular MSGD as the optimizer. This provides two benefits. First, there is no need to implement a new optimizer enhancing the applicability of the method. Second, the reweighting significantly reduces the computational cost of ENGM. To this aim, we use the estimated value for the norm of the gradients w.r.t. the input to normalize the gradients w.r.t. the network parameters indirectly by assigning a weight to the loss function computed on $\mathbf{x}_i$ as $\hat{L}_i := \hat{w}_i L_i$, where:

$$\hat{w}_i := \max(\frac{\alpha}{||\gamma_1 \nabla_{\mathbf{x}} L_i + \gamma_0||}, 1). \tag{5.7}$$

Here, optimizing the total loss $\frac{1}{|\mathcal{I}_t|}\sum_{i\in\mathcal{I}_t} \hat{L}_i$ using MSGD will approximately recover the functionality of ENGM on $\frac{1}{|\mathcal{I}_t|}\sum_{i\in\mathcal{I}_t} L_i$. To analyze the accuracy of estimating $\hat{w}_i$, we measure the average absolute value of the error during the training of the both models in AT and for three different values of $\alpha \in \{0.1, 1.0, 3.0\}$. Figures 5.3c and 5.3d visualize the error on two different datasets and network architectures. We observe that the maximum absolute value of error is less than 10% which advocates for the accuracy of estimating $\hat{w}_i$. For large values of α the error decreases during the training, while for small values of α the error increases. This points to a trade-off between the estimation error across the training process. It might be noted that the error is computed solely for AT since based on the evaluations in Figures 5.3a and 5.3d the correlation is stronger in NT.

Unlike $\nabla_\theta L_i$, $\nabla_x L_i$ can be computed in parallel for a batch of data using a single backpropagation. We consider two approaches for estimating γ_0 and γ_1 which result in two variations of ENGM. In the first approach, referred to as Approximated ENGM (A-ENGM), we evaluate $\nabla_\theta L_i$ for a single mini-batch every τ iterations and use moving average to update the latest estimate. Then for the intermediate iterations, we use the estimate values of γ to approximate the norm of gradients using Equation 5.6. In comparison, A-ENGM reduces the required number of additional backpropagations from $|\mathcal{I}_t|$ (for ENGM) to $1 + |\mathcal{I}_t|/\tau$. In practice, we observe that the interval, τ, for estimating γ values can be conveniently large as investigated in Section 5.3.4. Furthermore, we consider a second approach in which we simply set $\gamma_0 = 0$ and merge γ_1 into α. We refer to this approach as Naive ENGM (N-ENGM) which solely requires a single additional backpropagation.

5.3 Experiments and Analysis

We evaluate ENGM on three datasets of CIFAR-10, CIFAR-100 [95], and TinyImageNet [153]. Following the benchmark experimental setup for AT [133, 134, 146, 154], we conduct ablation studies and exploratory evaluations on ResNet-18 with 64 initial channels, originally developed for ImageNet. For SOTA evaluation, we use Wide ResNet-34 with depth factor 10 (WRN-34-10) [155].

Training Setup. Except for evaluations involving ENGM, all the models are trained using MSGD with momentum 0.9, weight decay 5×10^{-4} [144–146], batch size equal to 128, and initial learning rate of 0.1. The learning rate is decayed by 0.1 at epochs 75, 90, and the total number of epochs is set to 120 unless otherwise noted. The standard data augmentation including random crop with padding size 4 and horizontal flip is applied for all datasets. All input images are normalized to $[0, 1]$. Based on ablation studies in Section 5.3.4, we set α for ENGM, A-ENGM, and N-ENGM to 5, 5, and 0.5, respectively. The momentum for A-

ENGM is set to 0.7 based on empirical evaluations. PGD with 10 steps (PGD^{10}), $\varepsilon = 8/255$, and step size 2/255 is used as the attack to maximize the adversarial loss in ℓ_∞-norm ball. As suggested by Rice et al. [139], during the training we select the model with the highest robust accuracy against PGD^{20} with $\varepsilon = 8/255$ and step size $8/(255 \times 10)$ on a validation set of size $1,000$ as the best model. Only for PGD^{20}, we use margin loss instead of cross-entropy due to its better performance in evaluating the robustness of the model [156].

Evaluation Setup. We evaluate the model against two major attacks. First is the same PGD^{20} used in the training to find the best model. For a more rigorous evaluation of the robust performance, we follow the setup of the recent SOTA defense methods [131–133,145, 146,154,157,158] and use the benchmark adversarial robustness measure of AutoAttack (AA) [13]. AA has shown consistent superiority over other white box attacks such as JSMA [57], MIM [56], and CW [58]*. Both attacks in evaluations are applied on the test set, separated from the validation set. Maximum norm of perturbation, ε, is set to 8/255 and 128/255 for ℓ_∞-norm and ℓ_2-norm threat models. In addition to the robust accuracy, the robust overfitting of the model is computed as the difference between the best and the last robust accuracies (PGD^{20}) normalized over the best robust accuracy. All results are the average of three independent runs.

5.3.1 Comparison of Optimization methods

In this section, we evaluate and compare the proposed method with other possible choices for outer optimization in AT. As the fist baseline, we employ the conventional MSGD which is the optimizer in all of the previous AT methods. A popular and well-known trick to bound the gradient norm especially in recurrent neural networks is Gradient Norm Clipping (GNC) [159, 160]. GNC clips the gradient norm when it is greater than a threshold. This threshold is similar to α in our method. However, instead of bounding the gradient norm on each individual input example, GNC bounds the norm of the average gradients of the mini-batch. We consider the combination of MSGD with GNC as our second baseline and refer to it as MGNC. The clipping threshold α for MSGD+GNC is set to 25 based on empirical

*The benchmark is publicly available at *github.com/fra31/auto-attack*.

Optim.	Accuracy (%)				Overfit.
Method	Natural	Best	Last	AA	(%)
MSGD	**84.70**	50.87	44.15	46.77	13.2
MGNC	83.98	51.88	46.62	47.59	10.1
SNGM	83.73	51.95	46.80	47.75	9.9
F-ENGM	82.91	50.05	44.04	46.54	12.0
N-ENGM	84.36	52.19	48.79	48.06	6.5
A-ENGM	83.61	52.46	49.75	48.46	5.1
ENGM	83.44	**53.04**	**52.76**	**49.24**	**3.9**

Table 5.2: Comparison of ENGM with MSGD for outer optimization in AT (§5.3.1). 'Best' and 'Last' refer to the accuracy against PGD^{20} using the best and last checkpoints, respectively.

evaluations. SNGM, discussed in Section 5.2.3, is used as the third baseline. For our method, we compare the original ENGM with its accelerated versions, *i.e.*A-ENGM and N-ENGM. The coefficients α and τ for our methods are set to the best-performing values from Section 5.3.4. As an additional baseline, we develop another version of ENGM in which instead of bounding the norm of gradients, we normalize them to the constant value α, *i.e.*modifying Equation 5.5a to: $\mathbf{v}_{t+1} = \beta \mathbf{v}_t + \frac{1}{|\mathcal{I}_t|} \sum_{i \in \mathcal{I}_t} \frac{\nabla_\theta L_i}{||\nabla_\theta L_i||}$. We refer to this method as Fixed ENGM (F-ENGM).

Table 5.2 presents the results for these comparisons. We can see that the simple GNC enhances robust accuracy providing the same performance as SNGM. These improvements caused by simple modifications further confirms the negative effect of high gradient norm and variance on outer optimization in AT. ENGM consistently improves the robust accuracy over baselines. In addition, robust overfitting in ENGM is significantly lower than other baselines. This suggests that a major cause of robust overfitting in AT is the high fluctuation of gradients and the incompetence of MSGD in addressing it. The learning curves (robust test accuracy) for different optimization methods are depicted in Figure 5.5h. We observe that after the learning rate decay, the robust performance of ENGM and its variants does not deteriorate which confirms that they alleviate robust overfitting.

The best natural accuracy is provided by MSGD supporting the commonly observed

	AT	Optim.	Accuracy (%)			Overfit.
	Method	Method	Natural	PGD^{20}	AA	(%)
CIFAR-10	Vanilla	MSGD	**84.70**	50.87	46.77	13.2
		ENGM	83.44	53.04	49.24	3.9
	TRADES	MSGD	82.40	50.94	47.85	5.9
		ENGM	82.33	53.46	50.07	3.0
	MART	MSGD	83.68	51.05	48.29	6.1
		ENGM	83.03	53.56	50.48	4.6
	AWP	MSGD	82.98	52.55	50.12	4.6
		ENGM	83.10	**54.07**	**51.93**	**2.7**
CIFAR-100	Vanilla	MSGD	**57.75**	26.11	24.45	20.9
		ENGM	56.91	28.43	26.60	7.4
	TRADES	MSGD	56.00	29.04	26.93	10.6
		ENGM	55.65	30.68	29.20	7.1
	MART	MSGD	56.52	29.41	27.18	11.8
		ENGM	56.20	30.89	29.30	8.6
	AWP	MSGD	56.22	30.36	28.43	7.3
		ENGM	56.82	**31.24**	**30.46**	**6.3**
Tiny-ImageNet	Vanilla		35.71	7.47	6.92	26.37
		ENGM	29.78	11.29	8.54	10.10
	TRADES	MSGD	**37.26**	14.13	10.95	14.79
		ENGM	36.30	16.88	12.65	8.74
	MART	MSGD	37.06	13.79	10.08	15.94
		ENGM	36.53	16.90	12.99	8.20
	AWP	MSGD	36.13	16.29	13.09	10.67
		ENGM	36.81	**19.14**	**16.02**	**7.97**

Table 5.3: Comparison of MSGD and ENGM on different AT methods (§5.3.2). Note that ENGM consistently outperforms MSGD.

trade-off between the natural and robust accuracies [133, 161]. Table 5.1 presents the execution time for the optimization methods. The execution time of ENGM is roughly 8.5× longer than MSGD. However, A-ENGM and N-ENGM achieve notable speed-up and robust performance. As expected, the performance of A-ENGM is between N-ENGM (lower-bound)

Method	Optim.	Nat. Acc. (%)	AA (%)
ATES [132]	MSGD	86.84	50.72
BS [158]	MSGD	85.32	51.12
LBGAT [154]	MSGD	**88.22**	52.86
TRADES [133]	MSGD	84.92	53.08
MART [134]	MSGD	84.98	53.17
BERM [157]	MSGD	83.48	53.34
FAT [131]	MSGD	84.52	53.51
AWP [146]	MSGD	85.36	56.17
AWP	N-ENGM	85.40	57.11
AWP	ENGM	86.12	**57.45**

Table 5.4: Comparison of the benchmark robustness on WRN.

	Magnitude of Perturbation, ε $(\times\frac{1}{255})$					
	0	2	4	6	8	10
μ	4.25	5.10	6.09	7.73	10.04	14.21
σ^2	118.1	118.7	121.8	141.7	185.2	253.5
ρ_{MSGD}	0.33	0.41	0.57	0.93	1.15	1.24
ρ_{ENGM}	0.30	0.42	0.61	1.08	1.35	1.49

Table 5.5: Analyzing the impact of the perturbation magnitude on gradient properties and final robustness obtained by MSGD and ENGM (§5.3.2). AT with $\varepsilon = 0$ is equivalent to NT.

and ENGM (upper-bound) and is controlled by the estimation interval τ. Hence, we use N-ENGM and ENGM for the major evaluations to clearly compare the two performance bounds.

5.3.2 Combination with Benchmark AT Methods

In the this section, we incorporate the proposed optimization approaches into the benchmark AT methods including the vanilla method [43], TRADES [133], MART [134], and AWP [146]. Here, AWP represents the weight perturbation method applied on top of TRADES. The coefficient for the self-distillation loss in TRADES and MART is set to 6, and the maximum

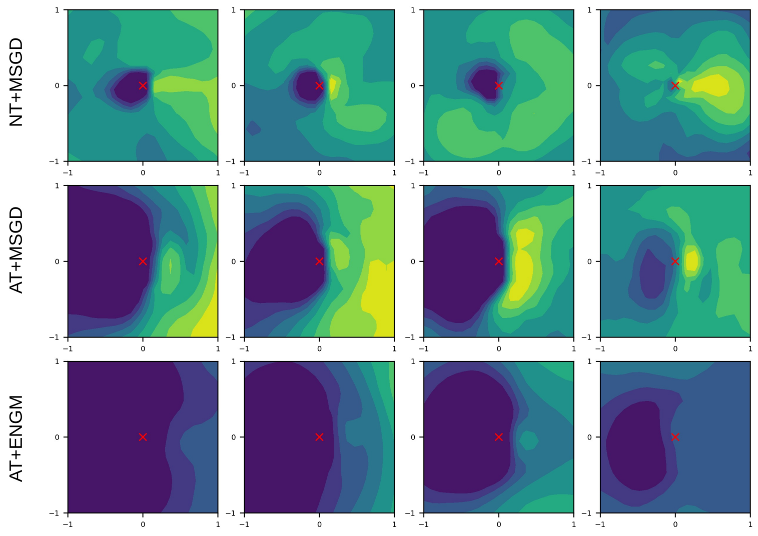


Figure 5.4: Visualization of the loss landscape on four examples from CIFAR-10 (§5.3.2). The cross mark denotes the input example. Loss level sets are equalized on each column.

magnitude of weight perturbation for AWP is set to 5×10^{-3}. The rest of the training setups are set to the best setup reported by the original papers. However, the total training epochs for all methods is set to 200 (learning rate decays by 0.1 at epochs 100 and 150) for the sake of consistency.

Table 5.3 presents the results for ℓ_∞-norm threat model. For results on ℓ_2-norm threat model please refer to Section 2 in Supp. material. We observe that ENGM consistently outperforms MSGD on robust performance. The average improvement in robustness against AA is 2.15% and 1.16% in ℓ_∞-norm and ℓ_2-norm, respectively. This suggests that the *amount* of perturbation in AT affects the convergence of the outer optimization. Consider the ℓ_2-norm as the unified metric, the amount of noise in ℓ_∞-norm threat model is roughly $3\times$ the norm of noise in the counterpart threat model. Combining these results with the evaluations in Figure 5.2 advocates that the improvement offered by ENGM over MSGD depends on the norm of perturbation. This observation is further investigated in Section 5.3.4.

AWP is previously shown to alleviate robust overfitting [146]. Interestingly, we find that TRADES and MART also reduce the robust overfitting independent of the optimization method. This suggests that the AT method can affect the robust overfitting. ENGM results in the lowest overfitting and consistently surpasses MSGD. On vanilla AT, replacing

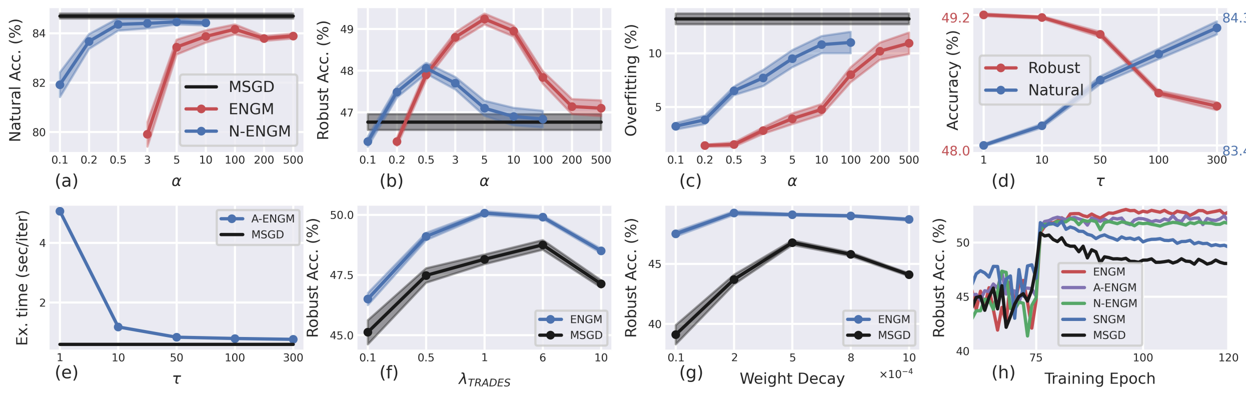


Figure 5.5: (a-g): Ablation studies on α, τ, λ_{TRADES}, and weight decay (§5.3.4). Note that α of ENGM scales to that of N-ENGM with $1/\gamma_1$. Robust accuracy is measured using AutoAttack [13]. (h): learning curves (robust test accuracy) for AT with different outer optimization methods. Results on last 60 epochs are plotted for better visualization of the robust overfitting. Robust accuracy is measured using PGD^{20}.

MSGD with ENGM results in 9.3%, 13.5%, and 16.2% reduction of overfitting on CIFAR-10, CIFAR-100, and TinyImageNet, respectively. These results advocate that, in addition to the AT method, the outer optimization method also affects the overfitting and limiting the sensitivity of the optimization method to the variance of the gradients can alleviate the robust overfitting.

As the last evaluation in this part, we visualize the loss landscape on networks optimized by MSGD and ENGM in Figure 5.4. This figure plots the loss values for the space spanned by the adversarial perturbation (PGD^{20}) and random noise, orthogonalized to the perturbation via Gram-Schmidt. We can see that ENGM results in a smoother loss landscape, known as an empirical evidence of the robustness [162]. This qualitative analysis further validates the effectiveness of ENGM for outer optimization in AT.

5.3.3 Comparison with SOTA

Here, we evaluate ENGM in the benchmark of AT, *i.e.*WRN-34-10 on CIFAR-10 dataset [131–133, 146, 154, 157, 158]. For training using ENGM, we set $\alpha = 10.4$ which is obtained by scaling the best α for ResNet-18 with the factor of 2.08, square root of the ratio of the total parameters of the two models (48.2M for WRN-34-10 vs. 11.1M for ResNet-18). To achieve SOTA performance, we consider AWP as the AT scheme. We train the model for 200 epochs with learning rate decay by 0.1 at epochs 100 and 150. The rest of the setting is the same as our previous evaluations. Table 5.4 presents the results for this experiment. AWP combined with ENGM and N-ENGM surpasses the previous SOTA by 1.28% and 0.94%, respectively. This validates the effectiveness of ENGM on large models. We also find that ENGM results in higher natural accuracy on AWP. This suggests that although AWP indirectly improves the outer optimization, its impact is orthogonal to ENGM.

5.3.4 Ablation Studies

We conduct ablation studies to investigate the impact of hyperparameters on the performance of ENGM and its two variants using ResNet-18 on CIFAR-10.

Impact of α: We measure the natural accuracy, robust accuracy (AA), and overfitting versus α. We conduct this experiment on ENGM/N-ENGM since they upper/lower bound the performance of A-ENGM.

Figures 5.5a, 5.5b, and 5.5c present the results for these evaluations. As expected, for large values of α all three values converge to that obtained by MSGD. Small values of α can be interpreted as training with a very small learning rate causing both the natural and robust accuracies to drop. Interestingly, we observe that the overfitting decreases significantly for small values of α. This confirms that the high variance of gradients in AT negatively affects the functionality of MSGD, *i.e.*ENGM with large α. We find that ENGM and N-ENGM achieve their optimal performance on ResNet-18 at α equal to 5 and 0.5, respectively. We select these as the optimal values for training the models in other experiments. Note that the optimal value of α is expected to be the same for ENGM and A-ENGM but different for N-ENGM. This is because the formulation of ENGM and A-ENGM is the same except that A-ENGM estimates the norm of gradients every τ iterations, and setting $\tau = 1$ recovers the exact ENGM. However, in N-ENGM, α is scaled by $1/\gamma_1$ according to the discussion in Section 5.2.5. The optimal α is scaled for other networks based on their capacity.

Impact of τ: We conduct experiments to evaluate the role of τ in AT setup with A-ENGM ($\alpha = 5$) as the optimizer and $\tau \in \{1, 10, 50, 100, 300\}$. It might be noted that each epoch in CIFAR-10 consists of 390 mini-batches of size 128. Hence, $\tau = 300$ is roughly equivalent to estimating the correlation at the end of each epoch. Figures 5.5d and 5.5e present the results for these evaluations. As expected, for small and large values of τ A-ENGM converges to ENGM and N-ENGM, respectively. For $\tau = 50$, obtained robustness is roughly 85% of the robustness obtained by ENGM while the training time is significantly lower (0.83 vs. 5.06) because the extra gradient computation is being performed every 50 iterations. Furthermore, we can see that τ controls the trade-off between the natural and robust accuracies.

Perturbation norm: As an initial exploration in this paper, we observed that AT induces higher gradient norm and variance. We also noticed in Section 5.3.2 that ENGM seems to outperform MSGD with a larger margin when the magnitude of perturbations is higher. Here, we further analyze the impact of the magnitude of perturbations on the gradient norm and variance induced by AT. This allows us to identify the extent of suitability of MSGD and ENGM for NT and AT. We train models in AT setup with ℓ_∞-norm threat model and varied size of perturbation, $\epsilon \in \{0, 2/255, 4/255, 6/255, 8/255, 10/255\}$. Both MSDG and ENGM are utilized for the outer optimization in these evaluations. We measure the average norm and variance of gradients across all training epochs. For a fair comparison, we compute the expected distance to the closest decision boundary as the unified robustness measure: $\rho := \mathrm{E}_{\mathbf{x}}[||\mathbf{x} - \mathbf{x}^*||]$, where $\mathbf{x}^*$ is the closest adversary to x computed using DeepFool [3].

Table 5.5 presents the results for this experiment. In NT (AT with $\epsilon = 0$), MSGD provides slightly better performance than ENGM. This is because in NT the norm and variance of gradients are naturally limited. As the ϵ increases, the expected norm and variance of the gradients also increase. This confirms our initial observation that AT induces higher gradient norm and variance. Consequently as expected, we find that in AT with larger magnitude of perturbations ENGM works better than MSGD.

Sensitivity to hyperparameters: One intriguing shortcoming of AT is sensitivity to hyperparameter setting. Several works have shown that a slight change in the modulus of the ℓ_2-norm regularization, *i.e.*weight decay, results in drastic changes in robust performance [144,145]. Here, we analyze the sensitivity of the proposed optimization method and compare it with that of MSGD. Figure 5.5g presents the results for this evaluation. We observe that ENGM exhibits significantly less sensitivity to changes in weight decay compared to MSGD. We hypothesis that high weight decay helps MSGD to prevent the bias from input examples with high gradient magnitude. ENGM achieves this goal by explicitly limiting the gradient magnitudes, and thus, is less sensitive to weight decay. We believe this phenomenon calls for more in depth analysis and defer it to future studies.

5.4 Conclusion

In this paper, we studied the role of outer optimization in AT. We empirically observed that AT induces higher gradient norm and variance which reduce the effectiveness of the conventional optimizer, *i.e.*MSGD. To address this issue, we developed an optimization method robust to the variance of gradients called ENGM. We provided two approximations to ENGM with significantly reduced computational complexity. Our evaluations validated the effectiveness of ENGM and its fast variants in AT setup. We also observed that ENGM alleviates shortcomings of AT including the robust overfitting and sensitivity to hyperparameters.

Chapter 6

Boosting Deep Face Recognition via Disentangling Appearance and Geometry

6.1 Introduction

Using the face as a biometric trait has several advantages for identification purposes. First, faces are naturally exposed and can be often captured remotely with suitable quality by incorporating a ubiquitous, moderately priced camera system. Second, the convenience of the acquisition procedure has promoted the acceptability of the modality compared to the fingerprint and iris which require direct cooperation of individuals. Third, the consistent morphological structure of faces, *i.e.*, semantic parts of the face share similar spatial properties among different individuals, also facilitates the process of reducing the variations of face images captured in unconstrained setups. In classical face recognition (FR) studies, the major challenge was to devise hand-crafted features that offer high inter-class separability and low intra-class variations [163–166].

The rapid development of technology over the last decade has had a profound impact on the performance and methodology of FR approaches. It has led to the generation of large-scale face datasets such as VGGFace [67], CASIA-WebFace [70], and MS-Celeb-1M [167]. Such comprehensive datasets of faces revealed detailed information regarding the manifold of

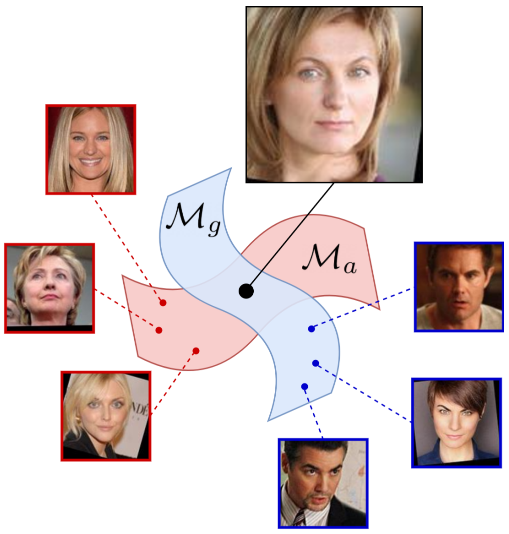


Figure 6.1: The proposed approach disentangles deep representations of the appearance and geometry of the face. $\mathcal{M}_\text{g}$ and $\mathcal{M}_\text{a}$ provide a schematic visualization of the manifolds of appearance and geometry constructed using our framework, respectively. Manifolds are superimposed at the input face representation.

natural faces and provided the supervision for training large-scale learning models. On the other hand, the development of parallel processing units has allowed an efficient optimization of DNNs which consist of millions of trainable parameters. Consequently, the classical problem of FR has transformed into the new challenge of finding efficient and powerful network architectures and suitable loss functions. To this aim, a myriad of approaches has been proposed to learn discriminative face representations using DNNs [30, 67, 68, 168, 169].

Most recently, spectacular performance of carefully designed network architectures, such as VGG [67] and ResNet [12], have concentrated attention on finding suitable loss functions and training criteria [74, 75]. A suitable criterion should force the model to learn discriminative representations for which the maximum intra-class distance is smaller than the minimum inter-class disparity [30]. However, the challenging effects of unconstrained environmental conditions, such as lightning and backgrounds, in addition to the intrinsic variations of human pose and facial expressions, complicates finding a suitable criterion. Contrary to the object recognition problem, in the FR task, the number of classes is extremely large and the number of available samples per each class is often limited and variable. This significantly degrades the performance of the well-established Softmax loss function, *i.e.*, the combination of the Softmax normalization and cross-entropy loss function. Indeed, Softmax loss yields separable features but cannot provide sufficient discrimination [30]. Pioneer deep learning-based FR approaches have sought to increase the discrimination power of deep representations by devising novel losses, such as contrastive loss [168], center loss [170] and triplet loss [68]. However, recent studies have demonstrated that considering angular distance instead of the euclidean distance for the Softmax loss significantly improves the discrimination power of representations [30, 169]. Hence, Softmax loss refined by angular distance has become the SOTA method for training deep models.

In this paper, we seek to improve the performance of deep FR models by considering a novel perspective: instead of modifying the classification loss functions to obtain compact and discriminative representations, we propose disentangling of the appearance and geometry of the face. The core idea of the paper is to construct geometrically identical faces by incorporating spatial transformations and exploiting their relative similarities to learn disentangled embedding representations. Practically, the disentanglement provides two benefits for the training procedure of deep FRs. First, it improves the generalization and training accuracy by geometrically augmenting the training set. Second, it enhances the learned knowledge of the early and intermediate layers of the deep model by enforcing them to satisfy the relative properties of appearance and geometry representations in the corresponding embedding spaces. We conduct extensive experiments to evaluate these benefits for the face recognition task and demonstrate that the knowledge learned through the disentangling approach can also be used to improve the performance of other face-related tasks, such as attribute prediction.

6.2 Related Work

Classical face recognition. Face recognition has always been an important computer vision problem due to its challenging aspects, such as the large number of classes and limited number of per-class samples. Classical approaches have mainly addressed the problem by finding discriminative hand-crafted representations for the face. Most of the attempts were strongly dependent on experimental observations since the prior knowledge needed for extracting hand-crafted features were scarce or hard to interpret. Besides, capturing large intra-class variations was also a major hurdle. Similar to the hierarchy of cascaded computations in DNNs, hand-crafted methods, such as LBP [163] and Gabor [171], compute local descriptors and combine them to obtain higher-level representations with more discriminative power. However, these heuristic methods can offer limited discrimination since they are not directly supervised to optimize the classification objective [172]. In addition, although they are data-independent, they cannot robustly capture intra-class variations.

Deep face recognition. In recent years, DNNs have achieved astonishing performance in face recognition which has gone even beyond human-level expertise. As the pioneers of the work, DeepFace [173] and DeepID [168] incorporated the well-known combination of Softmax normalization and cross-entropy loss for learning very deep representations of the face. These were accompanied by studies expanding the network architectures and gathering large-scale datasets, such as VGG [67]. Although Softmax loss provides separability of classes, the learned features are not discriminative enough. Hence, several novel training criteria and loss functions have been proposed to enhance the discrimination power of learned representations. Sun et al. [168] incorporated a verification loss to enhance the performance of the identification loss. Schroff et al. [68] proposed a novel training criterion called triplet loss in which the representations are forced to be discriminative based on the relations of a triplet of embedding samples. Wen et al. [170] proposed center loss to increase the intra-class compactness of representations. Finally, based on the observation that Softmax loss imposes an angular distribution on the representations, several studies have proposed to enhance the discrimination power of representations by mapping faces onto hyperspherical embeddings and measure their similarities using the Cosine measure [30, 169].

Disentangling geometry and appearance of the face. Geometry and appearance are the two main characteristics of the face which are highly correlated with the corresponding ID. Since the very first research on FR, the geometry is known to play a crucial role in identification [38,174,175]. This has also been exploited to find suitable hand-crafted features for face recognition [163]. Appearance is a major part of a general term called soft biometrics which encompasses all characteristics of individuals which do not need to be unique but can help recognize the ID, e.g. , hair color and gender. Several approaches have been considered in soft biometrics to enhance the FR performance [176, 177]. An important limitation of these studies is their dependence on the soft biometrics information in the dataset. They also require appearance information during the test phase.

Several prior studies attempted to disentangle the appearance and geometry of faces. Shu et al. [178] proposed an unsupervised approach by using a coupled autoencoder model. Each of the autoencoders is forced to learn the geometry or appearance representation of the input sample. The model provides the supervision for disentangling by reconstructing the original image using the combination of the two representations. Xing et al. [179] followed a similar methodology but incorporated variational autoencoders to enhance the performance of disentangling. These methods provided a novel insight toward the task. However, representations learned using autoencoders do not contain enough identification information to achieve SOTA performance in face recognition.

6.3 Disentangling Geometry and Appearance

In this study, the main supervision for disentangling appearance and geometry of faces is provided by constructing two pairs of face images. In the first pair, the appearance of faces is similar and the geometry is different, while, in the second pair, the geometry is similar and the appearance is different. For this purpose, we geometrically map an input face image to another ID in the training set using landmark information available for face alignment. The combination of the manipulated face image with its original version and the target face image construct the first and second pairs of faces, respectively.

6.3.1 Geometrically Identical Faces

Let $\mathrm{x_i}$ be an input face image belonging to class $\mathrm{y_i}$ and the set $\mathrm{l_i} = \{(\mathrm{u_j}, \mathrm{v_j}) : \forall \mathrm{j} \in \{1, \ldots, \mathrm{K}\}\}$ describe the 2D locations of K landmarks corresponding to $\mathrm{x_i}$. For each input face image, we find the closest neighbor face, $\mathrm{x_{i'}}$, from a different class, $\mathrm{y_{i'}} \neq \mathrm{y_i}$, in the geometry space

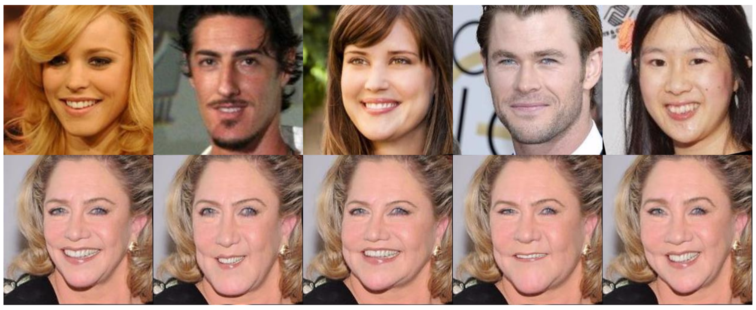


Figure 6.2: Examples of geometrically identical faces generated for five different IDs. First row shows input faces x_i, and the second row shows the corresponding face images $\hat{x}_{i'}$.

by computing i' as:

$$i' = \arg\min_j \frac{||l_i - l_j||_\infty}{\delta(y_i - y_j) + \varepsilon}, \tag{6.1}$$

where $\delta(.)$ is one when the input argument is not zero and is zero otherwise, and $\varepsilon \ll 1$, e.g. 10^{-6}. Here, ℓ_∞-norm assures that the selected neighbor face has a similar structure and pose as the input image in order to minimize the distortion caused by the spatial transformation in the next step. We assume that the rotation, scale, and translation of faces are aligned for the whole dataset, thus, the similarity of l_i and l_j can be measured in the same frame. It also worth mentioning that, for this work, we assume all landmark locations are vectorized before performing ℓ_p-norm, $||\cdot||_p$.

Although face image $x_{i'}$ has a geometry similar to x_i, their geometries do not completely match. To further match the geometry of two samples, we incorporate a spatial transformation and map $x_{i'}$ to x_i such that the resulting image has the same set of landmark locations. The deformed face image can be computed as:

$$\hat{x}_{i'} = T(x_{i'}, l_{i'}, l_i), \tag{6.2}$$

where T is the spatial transformation, *i.e.*TPS [72], which has a suitable capacity for the desired mapping compared to the affine transformation. The resulting face image, $\hat{x}_{i'}$, has the geometry of x_i and the appearance of $x_{i'}$. Figure 6.2 shows examples of this mapping. It may be noted that one can geometrically map all faces in the dataset to a canonical template in order to provide the supervision for decomposing the appearance and geometry of faces. However, computing a geometrically identical face for each input face provides two major

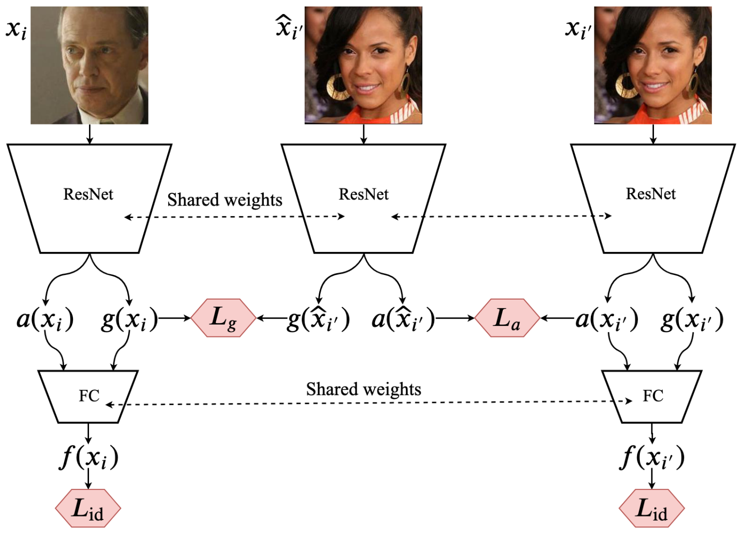


Figure 6.3: Schematic for training face recognition models enhanced by the proposed DAG approach.

benefits. First, it augments the training set by geometrically manipulating face images. Second, it increases the performance of the spatial transformer in matching the geometry of faces since each face is mapped to a face which is geometrically similar. In the next section, we use the geometric similarity and appearance disparity of $\mathrm{x_i}$ and $\mathrm{\hat{x}_{i'}}$ as the main supervision for the disentangling process.

6.3.2 Disentangling Networks

We define two networks for learning the discriminative representations of geometry and appearance of faces. Let $\mathbf{g} : \mathbb{R}^{\mathrm{w \times h \times 3}} \to \mathbb{R}^{\mathrm{d_g}}$ be the function mapping input image x to the geometric representation of the input face with the cardinality $\mathrm{d_g}$. Also, let $\mathbf{a} : \mathbb{R}^{\mathrm{w \times h \times 3}} \to \mathbb{R}^{\mathrm{d_a}}$ be the function mapping the same face to the representation of the appearance. For brevity, we assume the cardinality of both representations are equal $\mathrm{d_g = d_a = d}$. We also define a third function $\mathrm{f} : \mathbb{R}^{\mathrm{d}} \times \mathbb{R}^{\mathrm{d}} \to \mathbb{R}^{\mathrm{d'}}$ which takes the geometry and appearance representations and maps them to the final $\mathrm{d'}$-dimensional representation of the input face.

Based on the properties of geometrically identical faces defined in section 6.3.1, representations of the appearance and geometry of the faces $\mathrm{x_i}$, $\mathrm{x_{i'}}$, and $\mathrm{\hat{x}_{i'}}$ should satisfy following conditions: i) the geometry representations of the input face and the manipulated face should

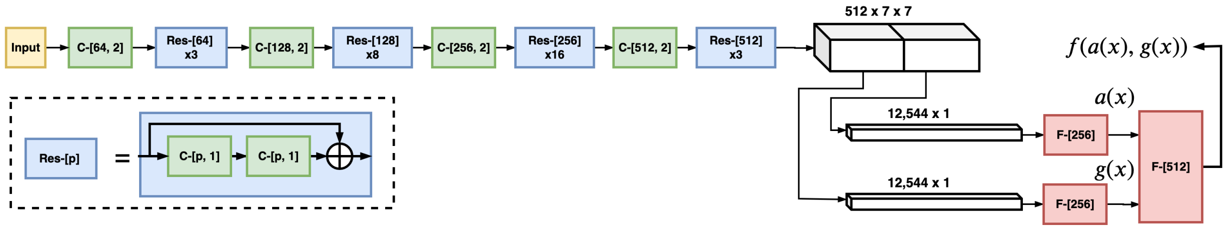


Figure 6.4: Architecture of the disentangling model. The network comprised of 3x3 convolutions consisting of p filters with the stride of s followed by PReLU (C-[p, s]), and fully-connected with p outputs (F-[p]). Each ResNet block consists of two consecutive convolutional layers followed by a shortcut from its input to its output.

be equal, $\mathbf{g}(x_i) \approx \mathbf{g}(\hat{x}_{i'})$, ii) the appearance representations of the neighbor face and its transformed version should be equal, $\mathbf{a}(x_{i'}) \approx \mathbf{a}(\hat{x}_{i'})$, and iii) integrating representations using f should provide enough information for an accurate identification of the input samples x_i and $x_{i'}$. We define proper loss functions to enforce such conditions on the representations. Intrinsically, the conventional Softmax loss function imposes an angular distribution on the learned representations [170]. Hence, we use the cosine similarity, which is a more suitable metric compared to Euclidean distance, to define the loss functions. As a result, representations of the appearance, geometry, and final ID in our framework follow an angular distribution.

We satisfy the first condition by defining a geometry-preserving loss function as:

$$\mathcal{L}_g := -\frac{1}{N}\sum_i \Phi(\mathbf{g}(x_i), \mathbf{g}(\hat{x}_{i'})) + \max(0, \Phi(\mathbf{g}(x_i), \mathbf{g}(x_{i'})) - \alpha_g \phi_g), \tag{6.3}$$

where $\Phi(v_1, v_2) = \frac{v_1^T v_2}{||v_1||||v_2||}$ computes the cosine similarity of input vectors, and N is the number of total samples in the batch. $\phi = \frac{||l_i - l_{i'}||_2}{||l_i - \bar{l}_i||_2}$ is a normalized measure of the distance of landmark locations, and $\bar{l}_i$ is the mean of landmark locations along two axes. α_g is a coefficient scaling ϕ_g to construct a margin which controls the angular distance between the geometry representation of x_i and $x_{i'}$. Indeed, Equation 6.3 forms an angular contrastive loss which aims to maximize the cosine similarity of $\mathbf{g}(x_i)$ and $\mathbf{g}(\hat{x_{i'}})$ while assuring that $\mathbf{g}(x_i)$ and $\mathbf{g}(x_{i'})$ are dissimilar, proportional to the landmark disparity of x_i and $x_{i'}$.

Similarly, we define the appearance-preserving loss function as:

$$\mathcal{L}_a := -\frac{1}{N}\sum_i \Phi(\mathbf{a}(x_{i'}), \mathbf{a}(\hat{x}_{i'})). \tag{6.4}$$

Face samples x_i and $x_{i'}$ are selected solely based on their geometric similarity, and their appearance can be completely different or very similar. Hence, Equation 6.4 does not consider a contrastive loss as the similarity of $\mathbf{a}(x_i)$ and $\mathbf{a}(x_{i'})$ is ambiguous. However, the identification loss function, described in the following, encourages appearance representations of different faces to take large enough distances for providing accurate identification.

So far, we developed a technique for **D**isentangling the **A**ppearance and **G**eometry (**DAG**) representations of face images. The final step is to combine this approach with conventional FR methods to establish highly discriminative representations. To this aim, we combine DAG with the family of A-Softmax [30,169] loss functions which have demonstrated significant performance for face recognition task. The main formulation of the loss function is:

$$\mathcal{L}_{id} = \frac{-1}{N}\sum_{i}\log\frac{e^{s(\cos(m_1\theta_{y_i,i})-m_2)}}{e^{s(\cos(m_1\theta_{y_i,i})-m_2)}+\sum\limits_{j\neq y_i} e^{s\cos(\theta_{j,i})}}, \tag{6.5}$$

where $z_i = f(\mathbf{a}(x_i), \mathbf{g}(x_i))$ is the final representation obtained by combining geometry and appearance information. Here, $\cos(\theta_{j,i}) = \frac{1}{||z_i||||W_i||}W_i^T z_i$, where W_i is the weight vector assigned to the ith class. Variables m_1 and m_2 are hyper-parameters controlling the angular margin, and s is the magnitude of angular representations. $\mathcal{L}_{id}$ with $(m_1 = 4, m_2 = 0, s = ||z_i||)$ and $(m_1 = 0, m_2 = 0.35, s = 64)$ denotes the loss functions defined in SphereFace [30] and CosFace [169], respectively. Later in Section 6.4.3, we combine both these loss functions with DAG to evaluate the effectiveness of the integrated model. The total loss for training the proposed framework is $\mathcal{L}_t = \mathcal{L}_{\overline{id}} + \lambda_a\mathcal{L}_a + \lambda_g\mathcal{L}_g$, where λ_a and λ_g are hyper-parameters scaling the appearance and geometry- preserving loss functions. Furthermore, $\mathcal{L}_{\overline{id}}$ is the average recognition loss function on x_i and $x_{i'}$. Figure 6.3 presents a schematic of the training criteria.

6.4 Experiments

Here, we evaluate the effectiveness of the proposed disentangling approach. First, we describe the implementation setup of the proposed model in section 6.4.1. Afterward, we conduct exploratory experiments to tune the hyper-parameters and provide some visualizations of the

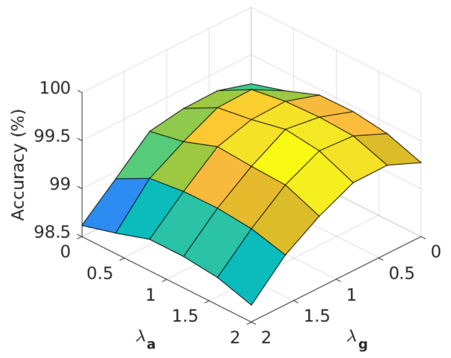


Figure 6.5: Accuracy (%) of Softmax loss enhanced by DAG trained with different values of λ_a and λ_g on LFW [7].

learned embedding representations in section 6.4.2. Finally, we evaluate the performance of the face recognition and attribute prediction tasks enhance by DAG in sections 6.4.3 and 6.4.7, respectively.

6.4.1 Implementation Details

Architecture and Hyper-parameters: We adopt ResNet [12] for the base network architecture of our model. To reduce the size of the model, the convolutional networks for extracting the geometry representation, $\mathbf{g}(x)$, and the appearance representation, $\mathbf{a}(x)$, are combined in a single ResNet-64 architecture. This network produces feature maps of spatial size 7×7 and the depth of 512 channels. Feature maps are then divided in depth into two chunks, and the first and second chunks are dedicated to the appearance and geometry, respectively. Feature maps are then reshaped to form vectors of size $12,544$ and passed to dedicated fully-connected layers to generate the final representations of the appearance, $\mathbf{a}(x)$, and geometry $\mathbf{g}(x)$. The cardinality of geometry and appearance representations is set to $d = 256$. The linear mapping f takes the concatenated outputs of $\mathbf{g}$ and $\mathbf{a}$ and maps them using a fully connected layer to the final embedding with the cardinality $d' = 512$. Figure 6.4 details the network architecture of the model.

The model is trained using Stochastic Gradient Descent (SGD) with the mini-batch size of 128 on four NVIDIA TITAN X GPUs. The initial value for the learning rate is set to 0.1 and multiplied by 0.9 in intervals of five epochs until its value is less than or equal to 10^{-6}.

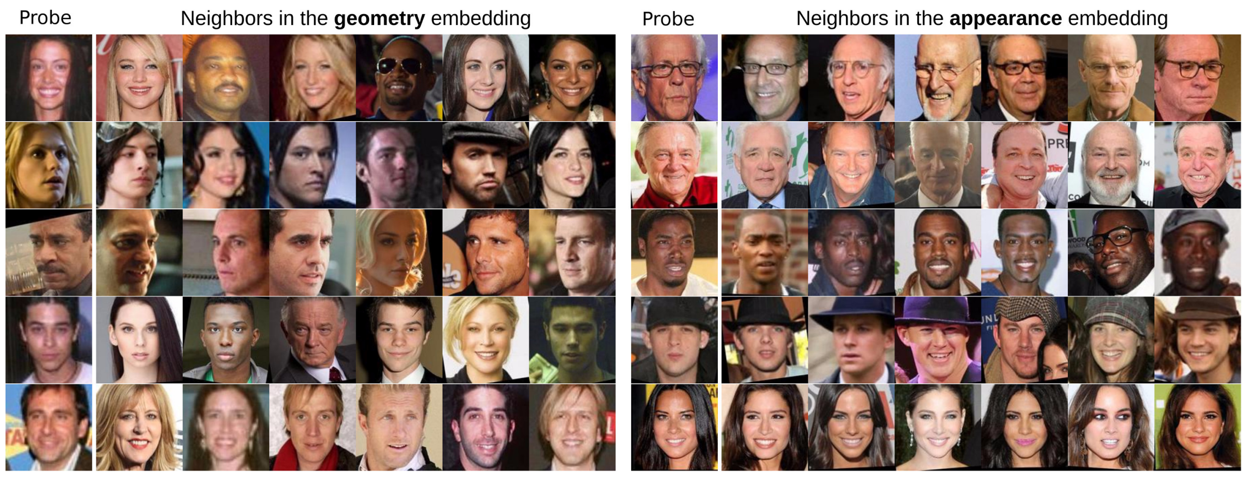

Probe
Neighbors in the **geometry** embedding
Probe
Neighbors in the **appearance** embedding

Figure 6.6: Visualizing the geometry (left block) and appearance (right block) embedding representations. For each probe sample, six nearest neighbors, based on cosine similarity in the embedding space, are demonstrated.

All models are trained for 600K iterations. The average of the landmark disparity measure ϕ_g on the training set of CASIA-WebFace [70] is ≈ 0.103. Accordingly to this value and based on practical evaluations, we found that $\alpha_g = 9.4$, *i.e.*keeping the angle of $\mathbf{g}(x_i)$ and $\mathbf{g}(x_{i'})$ greater than $\frac{\pi}{9}$, yields significant discriminability of geometry representations. We also set $\lambda_a = 1.3$ and $\lambda_g = 0.75$ based on experiments conducted in Section 6.4.2.

Preprocessing: Throughout the experiments, all faces are detected and aligned using DLib [71]. For each face, 68 landmark locations are extracted, and the closest neighbor in the geometry space is selected using Equation 6.1 over 1000 randomly selected face images from different IDs. Neighbor faces are then transformed spatially using Equation 6.2, and again aligned to compensate for the displacements caused by the spatial transformation. All face images are then resized to 112×112 and pixel values are scaled to $[-1, 1]$.

6.4.2 Exploratory Experiments

In this section, we first conduct experiments to evaluate the role of two hyper-parameters of DAG including λ_a, λ_g. Then, we perform a visualization experiment to demonstrate the effectiveness of DAG in learning rich geometry and appearance representations. We train a deep FR model using Softmax loss enhanced by DAG with different values of λ_a and

λ_g in the range $[0, 2]$, and evaluate the recognition performance on the LFW [7] dataset. Figure 6.5 presents the results for this experiment. The model reduces to a conventional face recognizer when λ_a and λ_g are zero. For large values of these parameters, the performance deteriorates which shows that the appearance and geometry loss functions dominate the identification objective. The maximum performance of the model occurs at $\lambda_a = 1.3$ and $\lambda_g = 0.75$. This confirms that DAG can enhance the performance of face recognition models. Furthermore, the performance of the model is more sensitive to λ_g compared to λ_a which shows that matching geometry representations of x_i and $\hat{x}_{i'}$ is harder than matching the appearance representations of $x_{i'}$ and $\hat{x}_{i'}$. We attribute this to the slight mismatch between the geometry of identical faces introduced because of the limited number of landmarks used to match the geometry of faces. On the other hand, the geometric transformation does not affect the appearance of faces. Hence, reducing the angular distance of appearance representations is more compatible with the identification loss.

Figure 6.6 presents a visualization of the embedding space representations learned by DAG. For this purpose, nearest neighbors of several probe faces are computed in the appearance or geometry embeddings based on their cosine similarity. Inspecting neighbor faces in geometry embedding suggests that DAG robustly captures geometry information of faces, such as relative distance and sizes of parts. Also, the large appearance variations of neighbors in the geometry embedding highlights that $\mathbf{g}(x)$ is invariant to the appearance. On the other hand, neighbors in the appearance embedding illustrate semantic appearance characteristics such as skin color, hair color, age, and gender. Interestingly, we observe that $\mathbf{a}(x)$ also captures appearance properties, such as the presence of eyeglasses and a hat, which are less prevalent compared to hair color and gender.

6.4.3 Face Recognition Enhanced by DAG

6.4.4 Performance of Combined Loss Functions

In this section, we combine DAG with several well-known face recognition methods and evaluate their performance on LFW, YTF, and MegaFace Challenge 1(MF1) [180]. We train models on the CASIA-WebFace [70] dataset with the same network architecture of modified

Method	LFW	YTF	MF1 Rank1	MF1 Veri.
Softmax [30]	97.89	93.1	54.88	66.31
Softmax+Aug.	98.15	94.5	58.90	70.02
Softmax+**DAG**	**98.58**	**94.7**	**60.73**	**71.62**
SphereFace [30]	99.40	94.9	73.19	86.38
SphereFace+Aug.	99.46	95.2	74.66	88.35
SphereFace+**DAG**	**99.55**	**95.6**	**75.28**	**88.90**
CosFace [169]	99.34	95.8	77.15	89.76
CosFace+Aug.	99.48	96.2	78.31	90.12
CosFace+**DAG**	**99.59**	**97.2**	**79.24**	**91.04**

Table 6.1: Evaluating the performance of well-known face recognition models enhanced with DAG. Verification refers to true acceptance rate under FAR= 10^{-6}.

ResNet-64 defined in Section 6.4.1. As discussed in Section 6.3.1, DAG utilizes geometrically transformed faces to disentangle appearance and geometry. These transformed faces augment the training set which potentially can improve the performance of deep face recognition. Hence, to better analyze the effectiveness of disentangling we consider an additional model trained on a quasi-augmented dataset. This dataset consists of around 1M images and formed by appending 10,575 subjects from MS-Celeb-1M [167] to CASIA-WebFace. The size of the quasi-augmented dataset is equal to the presumably augmented dataset constructed by the geometric transformation of DAG. Table 6.1 demonstrates the results for these experiments. As expected, training models on quasi-augmented dataset improves the performance. However, combining face recognition models with DAG consistently outperforms baselines. This suggests that disentangling the two major characteristics of faces enhances the training process of deep models and help them learn more abstract and representative features compared to the case when solely the training set is enlarged.

6.4.5 Benchmark Evaluations

For a fair benchmark comparison, we train the model on a large dataset of face images formed by combining VGGFace2 [69] and a private dataset. VGGFace2 contains 3.3M images from

Method	Training size	LFW	YTF
Deep Face [173]	4M	97.35	91.4
FaceNet [68]	200M	99.65	95.1
DeepFR [67]	2.6M	98.95	97.3
Baidu [181]	1.3M	99.13	-
SphereFace [30]	0.49M	99.42	95.0
CosFace [169]	5M	99.73	97.6
SphereFace+**DAG**	4M	99.67	96.2
CoseFace+**DAG**	4M	**99.81**	**98.0**

Table 6.2: Benchmark evaluation of face verification performance (%) on LFW and YTF.

9.1K identities with the average sample per identity of 362. The final dataset encompasses 4M images and 11.3K identities.

LFW and YTF. For evaluating the model on LFW, we follow the standard protocol of unrestricted with labeled outside data [7] and report our results on 6,000 pairs constructed using the test subset. YTF [183] consists of 3,425 videos of 1,595 unique IDs. Each video contains 181.3 frames on average which are downloaded from YouTube. Again, we follow the standard protocol of unrestricted with labeled outside data [7] and conduct experiments on 5,000 video pairs. Table 6.2 presents the results for these experiments. Integrating DAG with the well-known face recognition methods consistently enhances their performance on both LFW and YTF datasets.

MegaFace. We further evaluate the identification and verification performance of face recognition models enhanced by DAG using the challenging and large-scale benchmark of MegaFace [180]. MegaFace consists of a probe and a gallery subset. The gallery contains more than 1 million images from 640k individuals. The probe dataset is formed by combining FaceScrub [184] and FGNet datasets. The first dataset contains 100K images from 530 unique IDs, and the second dataset contains 1,002 images from 82 IDs. Several standard test scenarios are defined to evaluate the identification, verification, and pose invariance performance of methods under two main protocols, *i.e.*, small and large training sets. The protocol is considered small or large when the training set involves less than or greater than 0.5 million images, respectively. We also consider multi-patch models to measure the

Method	Protocol	Acc.	Veri.
SIAT_MMLAB [170]	Small	65.23	76.72
DeepSense-Small	Small	70.98	82.85
BeijingFaceAll V2	Small	76.66	77.60
GRCCV	Small	77.67	74.88
FUDAN_CS_SDS [182]	Small	77.98	79.19
SphereFace (1-patch) [30]	Small	72.72	85.56
SphereFace (3-patch) [30]	Small	75.76	89.14
CosFace (1-patch) [169]	Small	77.11	89.88
CosFace (3-patch) [169]	Small	79.54	92.22
SphereFace+**DAG** (1-patch)	Small	77.32	91.25
SphereFace+**DAG** (3-patch)	Small	78.83	92.24
CosFace+**DAG** (1-patch)	Small	79.18	91.46
CosFace+**DAG** (3-patch)	Small	**82.54**	**94.79**
Beijing FaceAll_Norm 1600	Large	64.80	67.11
Google-FaceNet v8 [68]	Large	70.49	86.47
NTechLab-facenx large	Large	73.30	85.08
SIATMMLAB TencentVision	Large	74.20	87.27
DeepSense V2	Large	81.29	95.99
Youtu Lab	Large	83.29	91.34
Vocord-deepVo V3	Large	**91.76**	94.96
SphereFace (1-patch) [30]	Large	77.44	91.49
SphereFace (3-patch) [30]	Large	80.85	93.60
CosFace (1-patch) [169]	Large	82.72	96.65
CosFace (3-patch) [169]	Large	84.26	97.96
SphereFace+**DAG** (1-patch)	Large	81.28	93.32
SphereFace+**DAG** (3-patch)	Large	85.76	94.87
CosFace+**DAG** (1-patch)	Large	85.62	97.26
CosFace+**DAG** (3-patch)	Large	87.02	**98.29**

Table 6.3: Performance of face identification and verification on MegaFace dataset. Verification measure (Veri.) denotes TAR at FAR $= 10^{-6}$.

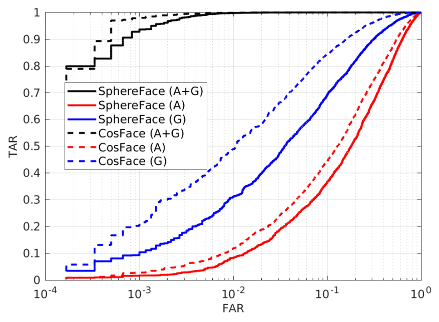


Figure 6.7: ROC curves for matching face images based on representations of appearance (A), geometry (G), and their combination (A+G) on LFW [7].

Method	$\mathbf{a}(\mathrm{x})$	$\mathbf{g}(\mathrm{x})$	$\mathrm{f}(\mathbf{a}(\mathrm{x}), \mathbf{g}(\mathrm{x}))$
SphereFace+**DAG**	67.03	81.12	99.55
CosFace+**DAG**	68.56	87.45	99.59

Table 6.4: Identification performance of individual representations on LFW [7].

performance of an ensemble of the proposed model based on the setup described in [30].

Table 6.3 summarizes the results for these evaluations. On both protocols, integrating DAG with SphereFace and CosFace enhances both the identification and verification performances. Particularly, on three out of four test setups, the integration with CosFace achieves superior performance compared to the previous approaches.

Method	Bald	Bangs	Big Lips	Big Nose	Black Hair	Blond Hair	Brown Hair	Bushy Eyeb.	Chubby	Eyeglasses	Male	Mustache	Narr. Eye	No beard	Oval Face	Smiling	Wear. Hat	Young
FaceTracer [185]	89	88	64	74	70	80	60	80	86	98	91	91	82	90	64	89	89	80
PANDA [186]	96	92	67	75	85	93	77	86	86	98	97	93	84	93	65	92	96	84
LNets+ANet [187]	98	95	68	78	88	95	80	90	91	99	98	**95**	81	**95**	66	92	99	87
Model$_A$	82.3	80.1	59.5	70.3	64.9	71.4	59.6	70.5	74.0	85.4	82.6	82.6	82.1	73.7	60.6	76.7	78.6	68.3
Model$_A$+**DAG**	89.4	86.6	67.6	77.7	75.4	79.3	66.9	79.2	87.7	94.2	90.4	89.3	85.0	79.2	68.1	84.2	92.5	80.7
Model$_B$	97.2	94.6	70.5	78.3	91.6	93.7	76.4	88.7	93.0	98.2	98.8	91.5	85.6	84.6	72.6	92.6	97.3	89.0
Model$_B$+**DAG**	**99.1**	**97.3**	**77.8**	**85.2**	**92.5**	**97.4**	**84.6**	**94.4**	**94.2**	**99.3**	**99.1**	93.2	**88.6**	91.6	**77.3**	**95.2**	**99.2**	**93.4**

Table 6.5: Performance comparison of facial attribute prediction methods on CelebA.

6.4.6 Performance of Individual Representations

In previous sections, we demonstrated that disentangling appearance and geometry representations of faces can enhance the recognition performance. We attribute this performance boost to the highly representative and complementary features extracted by each of the geometry and appearance branches. Indeed, forcing the model to learn disentangled embedding spaces helps the early and intermediate convolutional layers to extract more representative features. Here, we examine the appearance and geometry representations individually to evaluate their role in the recognition task. To this aim, we consider $\mathbf{g}(\mathrm{x})$, $\mathbf{a}(\mathrm{x})$, and $\mathrm{f}(\mathbf{a}(\mathrm{x}), \mathbf{g}(\mathrm{x}))$ for matching face images of LFW using the setup described in Section 6.4.5. Figure 6.7 and Table 6.4 present the result for this experiment. The geometry representations on both methods provide more informative representations for the identification task compared to appearance representations. This was expected since the geometry of faces contain rich discriminative information and appearance of faces has less variations intrinsically.

6.4.7 Knowledge Transfer for Attribute Prediction

Learning rich representations for faces can be beneficial for applications other than face recognition. Another important task related to the face is attribute prediction. Lack of both training data and variations in properties of faces in annotated datasets is the major factor deteriorating the performance of facial attribute prediction models [188, 189]. To address these problems, a major group of methods build their models upon representations learned from large-scale face recognition datasets [187]. In this section, we transfer the knowledge learned using a face recognition method integrated with DAG to evaluate its usefulness for attribute prediction. We conduct our experiments on the widely used face attribute dataset of CelebA [187] which contains 10,000 identities with around 200,000 samples. Eighteen major attributes are selected for comparing the results.

We use the exact model trained in Section 6.4.4 using the Softmax loss function and drop the last two fully-connected layers, *i.e.*preserving $\mathbf{a}(\mathrm{x})$ and $\mathbf{g}(\mathrm{x})$. Afterward, we define two test models, namely $\mathrm{Model}_{\mathrm{A}}$ and $\mathrm{Model}_{\mathrm{B}}$. In $\mathrm{Model}_{\mathrm{A}}$, we freeze $\mathbf{a}(\mathrm{x})$ and $\mathbf{g}(\mathrm{x})$, and train a fully-connected layer to map the learned representations to the final prediction of each attribute. Hence, this model mimics the weakly-supervised framework in which all layers except the last linear layer are trained solely using recognition supervision. In Model_B, we also fine-tune $\mathbf{a}(\text{x})$ and $\mathbf{g}(\text{x})$ using $0.1\times$ the learning rate of the fully-connected layers. For each attribute, a dedicated fully-connected layer is used to perform binary classification using the conventional softmax loss function. Fully-connected layers are optimized using SGD with the initial learning rate of 0.01 and the decay rate of 0.9 every four epochs. All models are trained for 40 epochs. We compare our results to FaceTracer [185], PANDA [186], and LNets+ANet [187]. Following the setup for FaceTracer and PANDA, we use the landmark information of faces to crop all faces.

Table 6.5 summarizes the results for this experiment. Transferring the knowledge learned by the face recognition models enhanced by DAG consistently improves the performance of attribute prediction. Particularly, for Model_A which is trained using a weakly-supervised scheme, integration of DAG with the original face recognizer improves the performance of attribute prediction by **8.34**% on average. This confirms that the DAG approach can help deep models to capture more informative representations of the face. Furthermore, fine-tuning the trained face recognizer enhanced by DAG using the attribute classification task outperforms baselines on 16 attributes out of 18. This also demonstrates that models enhanced by DAG can provide more sophisticated knowledge compared to conventional face recognition models.

6.5 Conclusion

In this paper, we propose the disentanglement of appearance and geometry representations for the face recognition task. We demonstrate that this approach boosts the deep face recognition performance by augmenting the training set and improving the knowledge learned by early and intermediate convolutional layers. Through extensive experiments, we validate the effectiveness of the proposed approach for face recognition and facial attribute prediction on challenging datasets. The individual capacity of the appearance and geometry representations are evaluated in additional experiments to analyze their semantic role in the face recognition task. Our results suggest that task-specific considerations for the training phase can further improve the performance of deep learning models.

Chapter 7

SuperMix: Supervising the Mixing Data Augmentation

7.1 Introduction

Despite the revolutionary performance of DNNs, they easily overfit when the training set is qualitatively or quantitatively deficient [190, 191]. Quality of the data can be interpreted as how well the data is expressive of the true distribution of inputs in the underlying task. This helps the model to learn discriminative patterns likely to occur at inference time. Quantity of the data, on the other hand, allows the model to observe discriminative patterns from different views and generalize the task-specific notions according to the major factors of variation in the input domain. Although analytical analysis of such important properties of the data has remained arduous [192], empirical evaluations on training deep models often highlight a common observation: incorporating more data leads to a better generalization [193, 194]. Hence, data augmentation has become a fundamental component of the training paradigms, aiming to enlarge the training set by transforming images in the given dataset.

Conventional image data augmentation involves combinations of context-preserving transformations, such as horizontal flip, crop, scale, color manipulation, and cut out [95, 195, 196]. Recently, notable efforts have been devoted to improving the augmentation, e.g. , by automating the search for the optimal augmentation policies [197–199]. The majority of these methods have focused on transforming single images, while ignoring the potentially

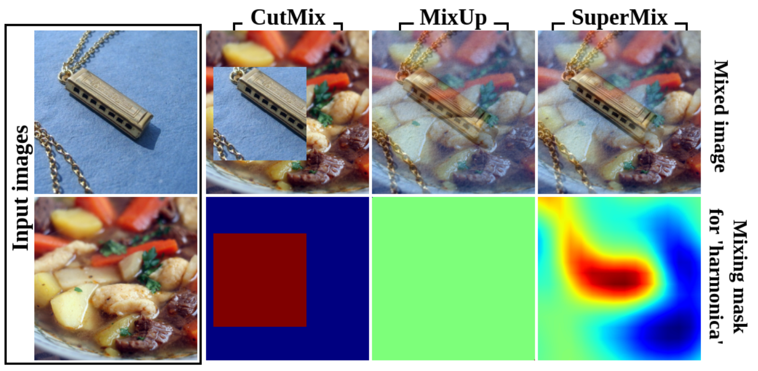


Figure 7.1: SuperMix combines salient regions in input images to construct unseen data for training.

very useful combination of multiple images for augmentation. To address this shortcoming, several studies have considered combining multiple images to construct novel images [44–46, 200, 201]. However, these methods either mix images blindly and disregard the salient regions [44–46, 202] or do not scale to large-scale problems [200]. Furthermore, the current mixing functions are not expressive enough and often suppresses visual patterns by averaging or covering features in one image with the trivial features in another image. The corresponding pseudo labels are also not accurate and constrain the training performance [202].

This paper presents a mixing augmentation approach termed SuperMix, which exploits the salient regions of input images to construct more advantageous mixed data. The supervision for this purpose can be obtained from the target model itself, *i.e.*, self-training [203–208], or a more sophisticated model aiming to guide a student network via knowledge transfer [209, 210]. Figure 7.1 provides a visual comparison of mixed images produced by different methods. In a nutshell, the contributions of the paper are as follows:

- We formalize the problem of supervised mixing augmentation using a set of mixing masks associating the pixel value at each spatial location in the mixed image to the spatial locations in the input images.
- The optimization problem is carefully constrained to assure that the solutions are rich in salient features and comply with the realistic image priors.
- We develop a modified Newton iterative algorithm for SuperMix, suitable for large-

scale applications. This approach provides 65× speed-up as compared to SGD on ImageNet.

- We demonstrate that mixed images intrinsically induce smooth predictions, and thus, help reveal knowledge of the teacher model in knowledge distillation.

7.2 Related work

Data augmentation: Data augmentation aims to improve the generalization of the model by enlarging the train set using transformations preserving the context of inputs in the learning problem. Conventional image transformations for this purpose are horizontal flip, crop, scale, color manipulation, and cut out [95, 195, 196]. A contemporary trend of research on the topic has focused on selecting the best sequence of transformations according to the task, dataset, and learning model. AutoAugment (AA) [197] automated the search for augmentation policies given a predefined set of transformations. Despite the significant performance of AA, it suffers from prohibitive training complexity imposed by Reinforcement Learning. Multiple approaches have attempted to reduce the training complexity by employing more efficient search methods, e.g. , density matching in fast AutoAugment (FAA) [198], or population based augmentation (PBA) [211]. RandAugment (RA) [199] have shown that the search space and selection criteria can be significantly simplified by carefully combining random transformations. However, these methods ignore the potentially useful combination of multiple images for augmentation.

Mixing augmentation: Several recent studies have considered employing multiple images for data augmentation [44–46, 200, 202]. Smart Augmentation [200] proposed merging multiple images from the same class using a DNN trained concurrently with the target model. However, training an additional deep model alongside every target model is resource exhaustive and severely limits the scalability of the approach for large-scale problems. Moreover, the method is restricted to merge images from the same class which limits the diversity and novelty of visual patterns in the merged images. MixUp [44, 45] combined a pair of images for the augmentation by convex linear interpolation. CutMix [46] proposed overlaying a cropped area of an input image on another image to augment the data. Although MixUp and CutMix have demonstrated notable improvements to the training of object recognition models, they suffer from major shortcomings. First, they often average or replace salient regions in one image with insignificant regions, e.g. , background, in another image. Second, due to the lack of supervision the labels computed for the mixed images are not accurate and limits the usefulness of the mixed images. However, SuperMix addresses these issues by extracting the salient regions of inputs and carefully combining them according to the realistic image priors and saliency-preserving constrains.

7.3 Supervised Mixing Augmentation

Given a training set $\mathcal{D} = \{(\mathrm{x_i}, \mathrm{y_i})\}_{\mathrm{i=0}}^{\mathrm{N-1}}$, mixing methods take a subset $\mathrm{X} \subset \mathcal{D}$ to produce the mixed image $\hat{\mathrm{x}}$ and the corresponding label $\hat{\mathrm{y}}$. A crucial property of mixed images is that they must reside close to the manifold of the training data since the goal of the mixing is to enlarge the support of the training distribution. Previous mixing methods [44, 45] have considered this requirement by employing operations that preserve local smoothness of images. MixUp [44,45] combines a pair of images $(\mathrm{x_i}, \mathrm{x_j})$ using convex linear interpolation as: $\hat{\mathrm{x}} = \mathrm{r}\mathrm{x_i} + (1-\mathrm{r})\mathrm{x_j}$, where $\mathrm{r} \sim \mathrm{Beta}(\alpha, \alpha)$ is a random mixing weight from the symmetric Beta distribution with $\alpha \in (0, \infty)$. Due to the lack of supervision, the soft label for $\hat{\mathrm{x}}$ is computed using the same linear interpolation as: $\hat{\mathrm{y}} = \mathrm{r}\delta(\mathrm{y_i}) + (1 - \mathrm{r})\delta(\mathrm{y_j})$, where $\delta(\cdot)$ is the one-hot encoding function. This blind mixing suffers from two shortcomings. First, coefficient r assigns an equal importance to the whole image which can suppress important features by averaging with the background or less important features from the other image. Second, the computed soft label, $\hat{\mathrm{y}}$, does not accurately describe the probability of classes represented by the mixed image and, thus, limits the effectiveness of the augmentation.

7.3.1 Mixing function

We formalize a general formulation for the augmentation function that allows multiple images to be combined locally. We use a set of mixing masks $\mathrm{M} = \{\mathrm{m_i}\}_{\mathrm{i=0}}^{\mathrm{k-1}}$, where $\mathrm{m_i} : \Lambda \rightarrow [0, 1]$ associates each spatial location $\mathrm{u} \in \Lambda$ in $\mathrm{x_i}$ with a scalar value $\mathrm{m_i(u)}$. Using the mixing

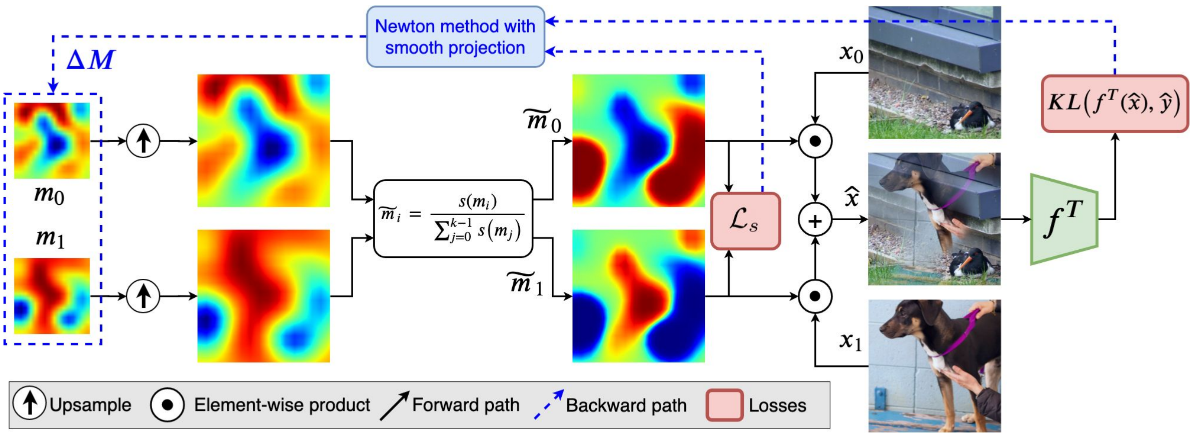


Figure 7.2: Schematic diagram of the proposed method for mixing $k = 2$ input images using the supervision from f^T.

masks, we define the mixing function as:

$$\hat{x} := \sum_{i=0}^{k-1} x_i \odot m_i, \tag{7.1}$$

where x_i is the ith sample in X, the operator $\odot$ denotes the element-wise product, and $\sum_i m_i(u) = 1$ to hold the convexity of the combination. The mixing function recovers MixUp [44] when $k = 2$ and all values in each mask are equal. It also recovers CutMix [46] when $k = 2$ and all values except the cropped area in one of the masks are equal to one. Figure 7.1 provides a visual comparison of the role of the masks in the mixing augmentation. In the next section, we describe how knowledge of a teacher model can be used to compute M such that the mixed image, $\hat{x}$, encompasses the rich visual information of images in X.

7.3.2 Supervised mixing

Let $f^T : \mathbb{R}^{W\times H\times C} \to [0,1]^n$ denote the probability vector predicted by the teacher (T) for n classes and f^T_i be the probability for the ith class. We optimize the set of masks M in Equation 7.1 such that all salient regions in X, according to the knowledge of the teacher, be present in the mixed image, $\hat{x}$. This can be interpreted as: $f^T(\hat{x}) \approx \hat{y}$, where $\hat{y}$ is high for classes associated with images in X. We formulate the target soft label, $\hat{y}$, computed in previous approaches [44,46] for $k = 2$ using the Beta distribution. We generalize for $k \geq 2$ by sampling the mixing coefficients from the Dirichlet distribution. Let $(r_0, \ldots, r_{k-1}) \sim \text{Dir}(\alpha)$

be a random sample from the symmetric multivariate Dirichlet distribution with parameter α and size k, we define the target soft label as:

$$\hat{y} := \sum_{i=0}^{k-1} r_i \delta\Big(y^T(x_i)\Big), \tag{7.2}$$

where $y^T(x_i) = \arg\max_j f_j^T(x_i)$ is the predicted class for $x_i \in X$, and $\delta(\cdot)$ is the one-hot encoding function.

The set of mixing masks can be optimized to minimize the divergence between the output of the teacher model on the mixed image and the target soft label computed in Equation 7.2. The masks must also hold two additional properties to comply with the realistic image priors. First, generated images must reside close the manifold of the training data. In practice, this interprets that each mask must be spatially smooth so that the generated images resemble the spatial structure of the inputs. Second, masks must be sparse across the input samples to ensure each spatial location in the output image is assigned merely to a single image which prevents averaging multiple images at each spatial location and suppressing important features. Considering these, the optimization problem for finding the mixing masks can be written as:

$$\begin{aligned} &\arg\min_{m_0,\ldots,m_{k-1}} \quad \mathrm{KL}(f^T(\hat{x})||\hat{y}) + \lambda_\sigma \mathcal{L}_\sigma(M) + \lambda_s \mathcal{L}_s(M) \quad \text{s.t.:} \\ &\qquad\qquad \text{a. } 0 \le m_i(u) \le 1, \quad \text{b. } \sum\nolimits_i m_i(u) = 1, \end{aligned} \tag{7.3}$$

where $\mathcal{L}_\sigma$ is a penalty term for the roughness of masks, e.g. , TV norm, $\mathcal{L}_s$ is a loss function to encourage sparsity of masks across input samples, and $\mathrm{KL}(\cdot||\cdot)$ is the Kullback-Leibler divergence.

Here, we provide an iterative algorithm to solve the optimization problem efficiently. At each iteration t, the convexity conditions can be satisfied by the following normalization:

$$\widetilde{m}_i^t = \frac{s(m_i^t)}{\sum_{j=0}^{k-1} s(m_j^t)}, \tag{7.4}$$

where $s(\cdot)$ is the sigmoid function. Hence, the generalized mixing function in Equation 7.1 takes the normalized masks to construct $\hat{x}$. Using the normalized masks, we define the sparsity promoting loss as:

$$\mathcal{L}_s := \tfrac{1}{kWH} \sum_{u,i} \widetilde{m}_i^t(u)\Big(\widetilde{m}_i^t(u) - 1\Big). \tag{7.5}$$

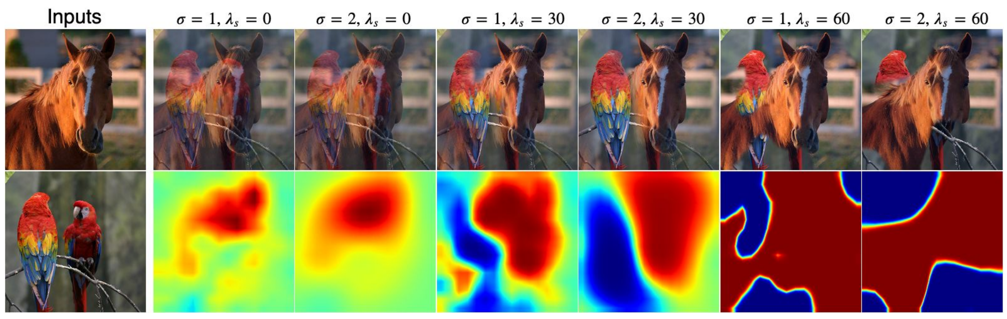


Figure 7.3: Visualizing the effect of smoothing factor, σ, and the sparsity promoting weight, λ_s, on the mixed images. Masks are estimated using ResNet34 and are associated with the 'horse' class.

This loss function encourages the mask values to approach 0 or 1. Since the values of masks at each spatial location sum to 1, due to the normalization in Equation 7.4, only one of the masks takes the high value to minimize the loss.

7.3.3 Optimization Method

A proper set of mixing masks can be estimated by minimizing the objective of SuperMix as $\mathcal{L}_{\text{SM}} = \text{KL} + \lambda_\sigma \mathcal{L}_\sigma + \lambda_s \mathcal{L}_s$. A reduced form of this problem has been studied in saliency detection and explanation of DNN predictions by employing SGD [91] or deep generators [212]. However, the current problem is more complex since multiple images are involved in the optimization and the roughness penalty and sparsity promoting loss should be minimized on all the corresponding masks. As we discussed and evaluated in Section 7.4.4, SGD is very slow and not feasible for solving the problem in case of large-scale image recognition tasks. Furthermore, employing a dedicated deep model to mix data by extending [212] makes the algorithm model-dependent and is not computationally efficient.

We develop a fast and efficient algorithm to optimize the mixing masks based on Newton's iterative method for finding roots of a nonlinear system of equations in the underdetermined case [3, 213]. Specifically, instead of optimizing $\mathcal{L}_{\text{SM}}$, we optimize $\mathcal{L}'_{\text{SM}} = \text{KL} + \lambda_s \mathcal{L}_s$ using a smooth projection (SP) [214] that directly satisfies the smoothness of masks. As we analyze later in Section 7.4.4, this significantly improves the execution time of the mixing. Considering the first-order approximation of $\mathcal{L}'_{\text{SM}}$ at M, each mask is updated at iteration t

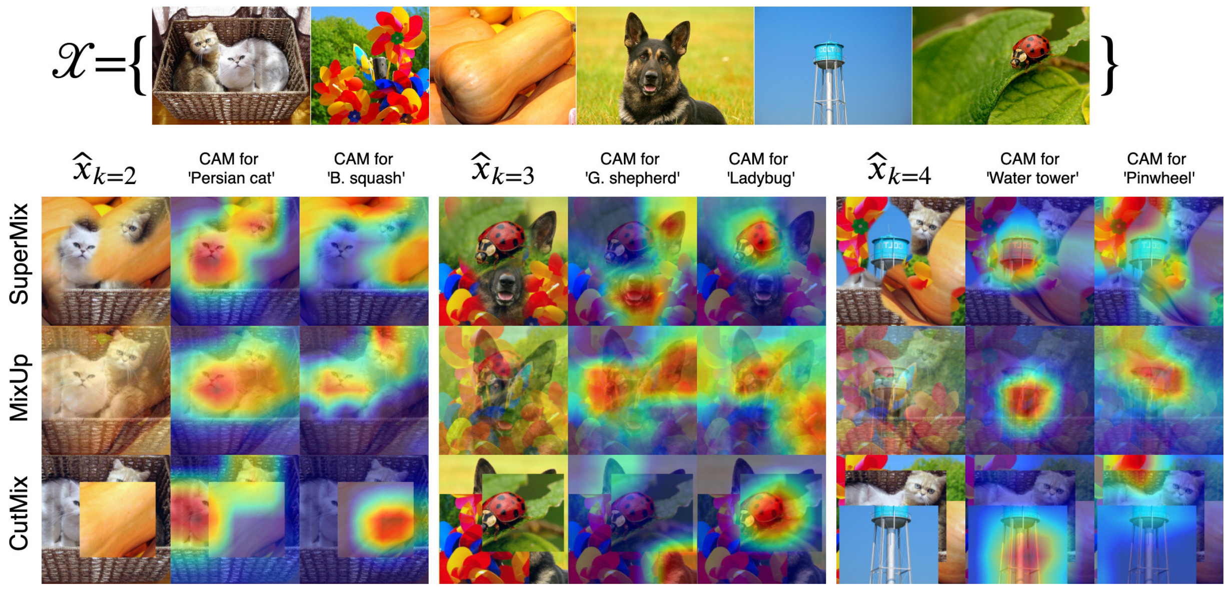


Figure 7.4: Visual comparison of the mixed images generated by SuperMix, MixUp, and CutMix, with k $\in \{2, 3, 4\}$ on ResNet34. Class activation maps [14] are computed for two classes in mixed images.

to find the roots as: $\mathrm{m}_{\mathrm{i}}^{\mathrm{t}+1} \leftarrow \mathrm{m}_{\mathrm{i}}^{\mathrm{t}} + \Delta \mathrm{m}_{\mathrm{i}}^{\mathrm{t}}$. Here, the update is computed using the Newton's method as:

$$\Delta \mathrm{M}^{\mathrm{t}} = \frac{-|\mathcal{L}'_{\mathrm{SM}}|}{||\nabla \mathcal{L}'_{\mathrm{SM}}||_2^2} \nabla \mathcal{L}'_{\mathrm{SM}}, \tag{7.6}$$

where the gradient is with respect to M^{t}, the concatenation of $\{\mathrm{m}_0^{\mathrm{t}}, \dots, \mathrm{m}_{\mathrm{k}-1}^{\mathrm{t}}\}$. Since both the divergence and $\mathcal{L}_{\mathrm{s}}$ are nonnegative, $|\mathcal{L}'_{\mathrm{SM}}| = \mathcal{L}'_{\mathrm{SM}}$. This formulation uses the ℓ_2-norm projection to compute $\Delta \mathrm{M}^{\mathrm{t}}$. We modify it using SP to preserve the smoothness of masks and compute the smooth update as:

$$\widetilde{\Delta \mathrm{M}}^{\mathrm{t}} = \frac{-\mathcal{L}'_{\mathrm{SM}}}{(\mathrm{g}_\sigma * \nabla \mathcal{L}'_{\mathrm{SM}})^{\mathrm{T}} \nabla \mathcal{L}'_{\mathrm{SM}}} (\mathrm{g}_\sigma * \nabla \mathcal{L}'_{\mathrm{SM}}), \tag{7.7}$$

where $\mathrm{g}_\sigma * \nabla \mathcal{L}'_{\mathrm{SM}}$ is a smoothed version of the gradients using the 2D Gaussian smoothing filter g with the standard deviation σ. It must be noted that all matrices in Equations 7.6 and 7.7 are vectorized before the matrix operations, and are reshaped back at the end of the iteration. In addition, due to the smoothness of masks, we optimize a down-sampled set of masks and up-sample them before performing the mixing. Algorithm 4 and Figure 7.2 demonstrate the detailed algorithm and schematic diagram for SuperMix, respectively.

Termination Criteria: The algorithm terminates when the Top-k predicted classes of $\mathrm{f}^{\mathrm{T}}(\hat{\mathrm{x}})$

Algorithm 4 SuperMix

```
1: inputs: Classifier $f^T$, set of k images X,
       low-pass filter $g_\sigma$.
2: output: Mixed sample $\hat{x}$.
3: $Y = \{\text{argmax}_j f^T_j(x_i) : x_i \in X\}$.
4: Sample $(r_0, \dots, r_{k-1})$ from $\text{Dir}(\alpha)$.
5: $\hat{y} = \sum_{i=0}^{k-1} r_i \delta(y^T(x_i))$.
6: Initialize $(m_0, \dots, m_{k-1}) \leftarrow 0$,
       $\hat{x}^0 \leftarrow \frac{1}{k}\sum_{x_i \in X} x_i$, $t \leftarrow 0$.
7: condition = Top-k predicted classes by $f(\hat{x}^t)$ are not in Y.
8: while condition do
9:     $\mathcal{L}'_{SM} = KL(f^T(\hat{x}^t)||\hat{y}) + \lambda_s \mathcal{L}_s$.
10:    $\widetilde{\Delta} M^t = \frac{-\mathcal{L}'_{SM}}{(g_\sigma * \nabla \mathcal{L}'_{SM})^T \nabla \mathcal{L}'_{SM}} g_\sigma * \nabla \mathcal{L}'_{SM}$.
11:    $m_i^{t+1} \leftarrow m_i^t + \widetilde{\Delta} m_i$ for $i \in \{0, \dots, k-1\}$.
12:    $\widetilde{m}_i^{t+1} = s(m_i^{t+1}) / \sum_{j=0}^{k-1} s(m_j^{t+1})$.
13:    $\hat{x}^{t+1} \leftarrow \sum_{i=0}^{k-1} x_i \odot \widetilde{m}_i^{t+1}$.
14:    $t \leftarrow t + 1$
15: end while
16: return $\hat{x}^t$.
```

are the same as the predicted class for samples in X. For instance, when X consists of two images recognized as 'cat' and 'dog', the Top-2 classes in $f^T(\hat{x})$ should be classes of 'cat' and 'dog'. This criterion assures that important features in the input set are visible in the mixed image. Figure 7.4 provides a visual comparison of the mixed images produced by different methods.

Dataset	Model	Base.	Automated aug.			Mixing aug.			SuperMix
			AA [197]	FAA [198]	RA [199]	MixUp	CutMix	SuperMix	+ RA [199]
CIFAR-100	WRN-40-2_a	74.0	79.3	79.4	79.2	77.2	77.9	**79.7**	79.9
	WRN-28-10	81.2	82.9	82.7	83.3	82.1	82.9	**83.6**	83.9
	S-S(26 2×96d)	82.9	**85.7**	85.4	85.6	84.8	85.0	85.5	85.8
ImageNet	ResNet-50	76.3/93.1	**77.6/93.8**	**77.6**/93.7	**77.6/93.8**	77.0/93.4	77.2/93.5	**77.6**/93.7	78.2/94.0
	ResNet-200	78.5/94.2	80.0/95.0	80.6/95.3	80.4/95.3	79.6/94.8	79.9/94.9	**80.8/95.4**	81.3/95.6

Table 7.1: Performance of augmentation methods on CIFAR-100 (Top-1 accuracy) and ImageNet (Top-1/Top-5 accuracy).

	Teacher	WRN-40-2$_b$		ResNet56	ResNet110		ResNet32x4	VGG13
	Student	WRN-16-2	WRN-40-1	ResNet20	ResNet20	ResNet32	ResNet8x4	VGG8
	Teacher acc.	75.61		72.34	74.31		79.42	74.64
	Student acc.	73.26	71.98	69.06	69.06	71.14	72.50	70.36
Distillation method	KD [210]	74.92	73.54	70.66	70.67	73.08	73.33	72.98
	CRD [215]	75.48	74.14	71.16	71.46	73.48	75.51	73.94
	CE+ ImgNet32	74.91	74.80	71.38	71.48	73.17	75.57	73.95
	CE+ MixUp	76.20*	75.53	72.00	72.27	74.60*	76.73	74.56
	CE+ CutMix	76.40*	75.85*	72.33	72.68	74.24	76.81	74.87*
	CE+ SuperMix	**76.93***	**76.11***	**72.64***	**72.75**	**74.80***	**77.16**	**75.38***
	KD+ ImgNet32	76.52*	75.70*	72.22	72.23	74.24	76.46	75.02*
	KD+ MixUp	76.58*	76.10*	72.89*	72.82	74.94*	77.07	75.58*
	KD+ CutMix	76.81*	76.45*	72.67*	72.83	74.87*	76.90	75.50*
	KD+ SuperMix	**77.45***	**76.53***	**73.19***	**72.96**	**75.21***	**77.59**	**76.03***

Table 7.2: Classification performance (%) of student models on CIFAR-100. Teacher and student are from the same architecture family but different depth/wideness and capacity. We denote by $\star$ results where the student surpasses the teacher performance. Only ImgNet32 uses unlabeled data from an external source. Average over 4 independent runs.

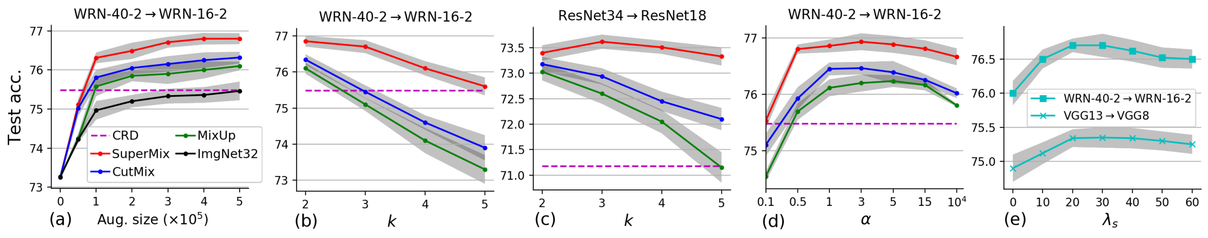


Figure 7.5: Evaluating the role of augmentation size and hyper-parameters.

7.4 Experiments

We evaluate the performance of SuperMix on two tasks of object classification and knowledge distillation [209, 210] using two benchmark datasets of CIFAR-100 [95] and ImageNet [96]. For knowledge distillation, we evaluate SuperMix on two major previous SOTA methods [210, 215] and two mixing augmentation techniques including MixUp and CutMix. For the sake of fair comparison, *pseudo labels for these blind mixing methods are computed using the same teacher employed in SuperMix.* All training experiments use random horizontal flip and random crop as the default augmentations. We perform the algorithm on random sets of input samples drawn from $\mathcal{D}$ to generate $\mathcal{D}'$. For the sake of brevity, we define the augmentation factor $\kappa = \frac{|\mathcal{D}'|}{|\mathcal{D}|}$ to show the ratio of the size of the mixed dataset over the size of the original dataset.

For knowledge distillation on CIFAR-100, we also consider an additional baseline by using unlabeled data from the training set of ImageNet32x32 [216] (ImgNet32) to construct unlabeled sets. This helps to better evaluate the role of the data provided by the mixing augmentation methods. We use SGD optimizer with an initial learning rate of 0.1 and momentum of 0.9. Weight decay is set to $5e-4$. The learning rate is decayed by 0.1 at epochs 200, 300, 400, and 500, and the maximum number of epochs is set to 600. Since in our experiments $\kappa \geq 1$, the number of epochs according to the mixed dataset will scale with $\frac{1}{\kappa}$ to keep the number of training iterations fixed for all experiments. For instance, when $\kappa = 5$, the maximum number of epochs for the mixed dataset is 120. The batch size is set to 128 and 256 for CIFAR-100 and ImageNet, respectively. For the CIFAR-100 dataset, we set σ of the Gaussian smoothing in SuperMix to 1 and the spatial size of the masks to 8×8. For ImageNet, σ is set to 2 and the size of masks is set to 16×16. For all benchmark comparisons, we set $\alpha = 3$ and $\lambda_s = 25$. Moreover, in all experiments, the performance of SuperMix is evaluated by generating 5×10^5 and 10^6 images on CIFAR-100 and ImageNet, respectively, unless otherwise noted. All the hyper-parameters for the distillation experiments are selected according to the experimental setup of [215] and the ablation studies in Section 7.4.3. Network architectures and settings for baseline methods are provided in the supplemental material.

7.4.1 Object classification

We follow the standard setup of evaluation for automated augmentation [197, 198, 211] and compare them with SuperMix on the task of object classification. For SuperMix, we first train the target model on the original dataset and then use it to generate mixed data with k equal to 2 and 3 for CIFAR-100 and ImageNet, respectively. Afterward, we train the target model from scratch on the mixture of the augmented data and the original data. Rest of the result are reported from the original papers. As an additional evaluation, we combine SuperMix with RangAugment (RA) [199]. For this purpose, we first mix images using SuperMix and then apply RA with the default parameters [199] for CIFAR-100 and ImageNet. Table 7.1 presents the results for these experiments. On four out of five experiments, SuperMix provide performance competitive to SOTA approaches of automated augmentation. Furthermore, combining RA with SuperMix further improves the performance of classification across all the experiments. These evaluations highlight the effectiveness of mixing multiple images for data augmentation.

7.4.2 Knowledge Distillation

In addition to KD [210] and CRD [215], we consider a simple method for distillation to highlight the effectiveness of mixing augmentation. In this method, we train the student models to classify mixed images labeled by the teacher model. The labels only show the winner class and does not contain any information regarding the rest of the classes. We refer to this method as Cross-Entropy (CE) distillation.

Results on CIFAR-100: Tables 7.2 and 7.3 presents the results for two challenging scenarios of distillation. In the first scenario, teacher and student are from the same family of architectures but have different depth/wideness and capacity. In the second scenario, teacher and student are from completely different network architectures. Employing the simple CE method using the mixed data consistently outperforms previous methods in both distillation scenarios. The data generated by SuperMix demonstrates the best performance across all evaluations, and, on five out of seven teacher-student setups from the same architecture family, students trained on the SuperMix data outperform their teachers. Last four rows in Tables 7.2 and 7.3 present the results for knowledge distillation using the original KD [210]. More importantly, results on MixUp, CutMix, and SuperMix demonstrate that they can notably enhance the performance of the distillation techniques.

These observations highlight three crucial points. First, the limited size of the training set is a major factor constraining the performance of knowledge distillation. According to Table 7.2, almost all of the students achieve comparable results to CRD when external data of ImgNet32 is provided. Second, mixing augmentation provides more informative data for distillation compared to unlabeled data from an external source. Third, the supervised mixing results in rich images that are highly favorable for knowledge distillation and outperforms blind mixing methods.

	Teacher	VGG13	ResNet50		ResNet32x4		WRN-40-2
	Student	MobileNetV2	MobileNetV2	VGG8	ShuffleNetV1	ShuffleNetV2	ShuffleNetV1
	Teacher acc.	74.64	79.34		79.42		75.61
	Student acc.	64.60	64.60	70.36	70.50	71.82	70.50
Distillation method	KD [210]	67.37	67.35	73.81	74.07	74.45	74.83
	CRD [215]	69.73	69.11	74.30	75.11	75.65	76.05⋆
	CE+ ImgNet32	68.85	68.01	73.96	76.80	77.56	75.87⋆
	CE+ MixUp	71.13	71.71	75.41	78.16	78.84	77.29⋆
	CE+ CutMix	70.93	70.64	75.84	77.89	79.32	77.50⋆
	CE+ SuperMix	**71.65**	**72.13**	**76.07**	**78.47**	**79.53⋆**	**77.92⋆**
	KD+ ImgNet32	69.14	68.44	74.32	76.87	77.90	76.23⋆
	KD+ MixUp	71.29	71.99	75.59	78.22	79.14	77.44⋆
	KD+ CutMix	71.10	70.93	76.01	77.92	79.53⋆	77.65⋆
	KD+ SuperMix	**71.81**	**72.40**	**76.28**	**78.51**	**79.80⋆**	**78.07⋆**

Table 7.3: Classification performance (%) of student models on CIFAR-100. Teacher and student models are from different architectures. We denote by ⋆ results where the student surpasses the teacher performance. Average over 4 independent runs.

Results on ImageNet: We showcase the effectiveness of the mixed data on ImageNet by distilling the knowledge of ResNet-34 into ResNet-18. Table 7.5 presents the results for the distillation on the ImageNet dataset. Using the simple CE method consistently outperforms the previous SOTA approaches. In five out of eight experiments of distillation using mixed images, the student outperforms the teacher. This demonstrates the scalability and effectiveness of the mixing augmentation for the task of knowledge distillation. Moreover, combining mixed data with the original distillation objective further enhances the distillation performance validating the effectiveness of the mixing augmentation for knowledge transfer in large-scale datasets.

7.4.3 Ablation studies

Impact of the size of the training set: In this part, we investigate how the size of the dataset affects the distillation performance by measuring the Top-1 test accuracy of WRN-16-2 versus the augmentation size on CIFAR-100. For all the mixing methods, we set $\mathrm{k} = 2$ and $\alpha = 1$, *i.e.*, sampling mixing coefficients from the uniform distribution. Figures 7.5a presents the results for these evaluations. The distillation performance improves by increasing the augmentation size and plateaus at 5×10^5. All the datasets generated using mixing augmentations outperform the unlabeled dataset of ImgNet32. This highlights the superiority of mixed images for knowledge transfer compared to unlabeled data from an external source. Based on these observations, we set the size of the mixed dataset to 5×10^5 for all experiments on CIFAR-100.

Impact of k: We evaluate the role of k by conducting experiments on CIFAR-100 and ImageNet datasets. Figures 7.5b and 7.5c present the results for this evaluation. A major shortcoming of MixUp and CutMix is that they mix images without any supervision. Including more input images to produce a mixed image increases the chance of incorrect cropping in CutMix, and averaging overlapping features in Mixup. This explains the notable deterioration of the distillation performance in all experiments with $\mathrm{k} > 2$ using these augmentation methods. Both of these incidents degrade the quality and effectiveness of features in the mixed image, which can also be observed from the visual comparisons provided in Figure

	Net		SGD	Newton	
				w/o SP	w/ SP
ImgNet	VGG16	ET(sec.)	15.41	6.59	**0.23**
		iters	34.5	15.1	**0.5**
	Res34	ET(sec.)	4.25	1.98	**0.06**
		iters	23.6	11.7	**0.3**
CIFAR	VGG13	ET(ms.)	482	97	**5**
		iters	19.5	3.7	**0.2**
	WRN	ET(ms.)	509	122	**6**
		iters	21.8	4.6	**0.2**

Table 7.4: Comparison of execution time.

7.4. We observe that the spatial size of the image can limit k. Performance of distillation using SuperMix degrades for k > 2 on CIFAR-100. However on ImageNet, k = 3 yields the best distillation performance.

Impact of α: Parameter α determines the probability distribution for the presence of each input class in the mixed image. We measure the performance of distillation versus several values of α to identify its optimal value. Figure 7.5d presents results for these experiments. For $\alpha \to 0$, the mixing augmentation becomes inactive since only one input category will appear in the augmented images, *i.e.*, $\mathrm{r}_0 = 1$ or $\mathrm{r}_1 = 1$. For $\alpha \to +\infty$, the contribution of images become equal, *i.e.*, $\mathrm{r}_0 = \mathrm{r}_1 = 0.5$. This is more favorable for distillation since both input images contribute equally to the mixed image. For $\alpha = 1$, contribution of each input in the mixed image is selected from the uniform distribution $\mathrm{Unif}(0, 1)$. According to the figures, we select $\alpha = 3$ for all other experiments unless otherwise noted.

	Teacher	Student	KD	CRD	CE	KD	CE	KD	CE	KD	CE	KD
					+MixUp$_{k=2}$		+CutMix$_{k=2}$		+SuperMix$_{k=2}$		+SuperMix$_{k=3}$	
Top-1	73.31	69.75	70.66	71.17	73.03	73.29	73.18	73.33*	73.42*	73.62*	73.65*	**73.83***
Top-5	91.42	89.07	89.88	90.13	91.27	91.44	91.36	91.44*	91.51*	91.66*	91.67*	**91.82***

Table 7.5: Top-1 and Top-5 classification accuracy of ResNet18 on ImageNet dataset. Results where the student surpasses the teacher performance are marked by $\star$. Average over 4 independent runs.

Sparsity among masks: The sparsity promoting loss forces each spatial location in the output image to be assigned to only one image in the input set. This improves the mixing performance by preserving the most important features in each spatial location. We evaluate the performance of distillation versus λ_s in Figure 7.5e. By increasing the weight of sparsity the performance of distillation improves until $\lambda_s \approx 30$. After that the accuracy of masks degrades since the sparsity promoting loss dominates the KL loss. Figure 7.3 evaluates this phenomenon by visualising the mixing mask versus λ_s.

7.4.4 Execution time

Here, we compute the execution time of SuperMix. To this aim, we define two baselines for the sake of comparison. For the first baseline, we use SGD instead of the Newton method to optimize the set of masks. The second baseline is the Newton method without SP. Hence, the optimization in both baselines is performed on $\mathcal{L}_{SM} = \mathrm{KL} + \lambda_\sigma \mathcal{L}_\sigma + \lambda_s \mathcal{L}_s$. Inspired by the previous work on saliency detection [91], we use the TV norm for the spatial smoothness loss as: $\mathcal{L}_s = \frac{1}{kWH} \sum_i \sum_{u \in \Lambda} ||\nabla m_i(u)||_3^3$. Based on experimental observations, we set $\lambda_s = 250$, learning rate of SGD to 0.1. All other parameters are set to the values identified in previous sections. All algorithms are implemented with parallel processing on two NVIDA Titan RTX with batch size of 128. For further implementation details, please refer to the released code.

Figure 7.4 presents the results for these comparisons. Newton method with SP, *i.e.*, SuperMix, is at least **65**$\times$ faster than SGD on both datasets. Moreover, due to SP which directly satisfied the spatial smoothness condition, SuperMix is at least **19**$\times$ faster than the same algorithm when it has to include $\mathcal{L}_s$.

7.4.5 Embedding space evaluations

We perform two sets of evaluations on CIFAR-100 to further analyze characteristics of the mixed images. In the first set of experiments, we feed the original data and the mixed images to VGG13 and visualize the output of the logits layer, in 2D for three random classes using PCA. The SuperMix images are generated with $k = 2$. Figure 7.6 demonstrates these evaluations. Representations for the SuperMix data has less overlap with the distribution of the

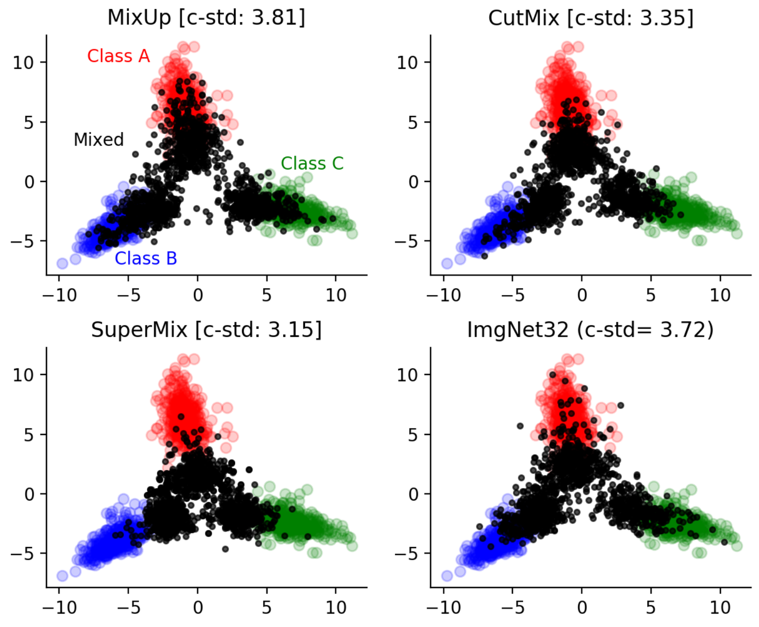


Figure 7.6: Visualizing representations for the mixed images.

representations for the original data. This suggests that the SuperMix data encompass more novel structure compared to the original data, unlabeled data from other mixing methods or an external source. The SuperMix data are harder to classify for the model since the representations are concentrated close to the center of the embedding. To better evaluate this, we compute the class standard deviation (c-std) of representations for each class. The computed values are reported on the top of the corresponding images in Figure 7.6.

Hinton et al. [210] pointed that smoothing out the predictions of a model can better reveal its knowledge of the task. Since SuperMix generates images by combining multiple inputs, the outputs of the model on SuperMix data are intrinsically more smooth compared to that of the other augmentation types. We validate this by computing the average of the sorted Top-5 probability predictions of VGG13 on the original and augmented images of CIFAR-100. As demonstrated in Figure 7.7, predictions of the target model is significantly smoother on mixed images. Moreover, SuperMix produces the data with the most smooth labels.

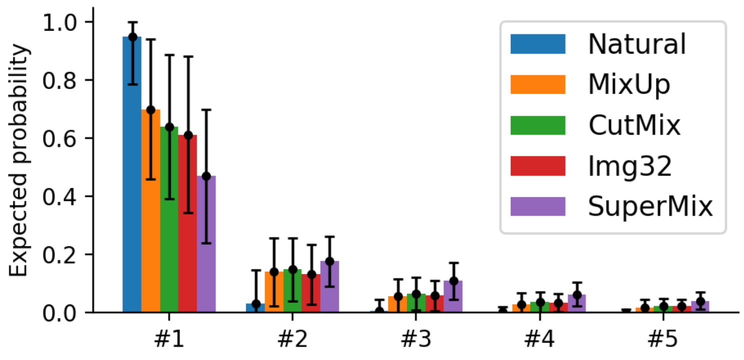


Figure 7.7: Distribution of top 5 predictions.

7.5 Conclusion

In this paper, we studied the potential of mixing multiple images using supervision of a teacher for the data augmentation. We proposed SuperMix, a supervised mixing augmentation method that combines salient regions in multiple images to produce unseen training samples. The effectiveness and efficiency of SuperMix is validated through extensive experiments, evaluations, and ablation studies. Specifically, incorporating SuperMix data for distillation enhances the SOTA methods of knowledge distillation. SuperMix provides comparable performance to the automated augmentation methods, and when combined, notably improves the generalization of the model.

Chapter 8

Fingerprint Distortion Rectification using Deep Convolutional Neural Networks

8.1 Introduction

The fingerprint is one of the most important biometric modalities due to its uniqueness and easy acquisition process. Leveraged by rapid advances in sensor technologies and matching algorithm development, automatic fingerprint recognition has been widely adopted as a highly-accurate identification method. The operation of a typical fingerprint recognition system consists of three main steps. In the preprocessing step, a raw fingerprint is enhanced to reduce noise, connect broken ridges and separate joined ridges. In the second step, exact ridge patterns are processed to extract local features, namely minutiae, from the enhanced image. In the final step, a match score between two fingerprint features is calculated by analyzing properties of minutiae (location, orientation, etc.) using local and global relationships between them.

In past decades, algorithms for fingerprint matching have advanced rapidly, resulting in the development of numerous and varied commercial fingerprint recognition systems. These algorithms have very high performance in identifying clean samples [217], but often fail in identifying samples which are distorted. Consequently, recognizing dirty fingerprints is a challenging problem for fingerprint recognition systems. Most of the fingerprint matching algorithms are based on calculating the relative properties between features within a fingerprint, and matching them with other fingerprints. However, distortion that can occur during the collection process changes the relative properties of fingerprint features and causes a notable decrease in recognition performance [218].

There are two main types of recognition scenarios. In the positive recognition scenario, the goal is user authentication, wherein the user cooperates with the recognition system in order to be recognized and obtain access to locations or systems. In contrast, the negative recognition scenario deals with an uncooperative user who is unwilling to be identified. Based on the recognition goal, the quality of the fingerprint can lead to different consequences. In the positive recognition scenario, low-quality fingerprints prevent legitimate users from being authenticated. Although this brings inconvenience, users learn to reduce distortion after several authentication attempts. Serious consequences of low-quality fingerprints are tied with the negative recognition scenario in which users may deliberately decrease the quality of fingerprint to avoid being identified [219]. Actually, attempts of altering and damaging fingerprints in order to impair identification have been reported by law enforcement officials [220, 221]. Hence, increasing fingerprint quality is a necessary task in negative recognition systems. Additionally, it provides the added benefit of reducing the inconvenience of false rejection of valid users in positive recognition systems.

The quality of fingerprint samples can be deteriorated by many factors, either geometrically or photometrically. The primary cause of photometric degradation is artifacts on the finger or sensor, such as oil, moisture or markings from previous impressions. Photometric degradation in fingerprints has been widely investigated in terms of detection [222–224] and compensation [225–229].

Fingers have cylindrical shape with relatively small radius compared to ridge pattern size. Capturing fingerprint samples is a complex mapping from a 3D surface to a 2D image, since the finger is being pressed onto a platen on a sensor. This mapping differs for each impression, referred to as geometric distortion. Geometric distortion is related to mechanical properties, such as the force and torque a user applies to the finger in the acquisition process. Different from photometric distortion, geometric distortion introduces translational and rotational error in the relative distances and orientations of local features. These relative distances and orientations of local features are the abstract identifiers of a user. In the presence of photometric distortion, the match score decreases since many minutiae may be missing, or false minutiae may be detected. On the contrary, in cases of severe geometric distortion, the match score decreases because the new composition of minutiae forms a completely different ID caused by the distortion. The issue is more critical in negative recognition systems, since distorted samples are still of high quality compared to clean samples, but matching algorithms fail to recognize them.

In this paper, we address the geometric distortion problem of fingerprint recognition systems by proposing a fast and effective distortion estimator which captures the non-linear properties of geometric distortion of fingerprints. While recently proposed methods handle distortion using a dictionary of distorted templates, for this work, we use a DCNN to estimate the principal distortion components of input samples. Our approach has the following contributions:

- There is no need to estimate the ridge frequency and orientation maps of input fingerprints.
- Distortion parameters are being estimated continuously to achieve more accurate rectifications.
- A notable decrease in rectification time due to embedding distortion templates in network parameters.

The rest of the paper is organized as follows. In section 8.2, related works are reviewed. Section 8.3 describes the proposed approach, and section 8.4 presents the experimental results. Finally, we conclude the paper in section 8.5.

8.2 Related Work

Various approaches have been proposed in the literature to tackle the issue of geometric distortion in fingerprints. Designing specific acquisition hardware which detects distortion

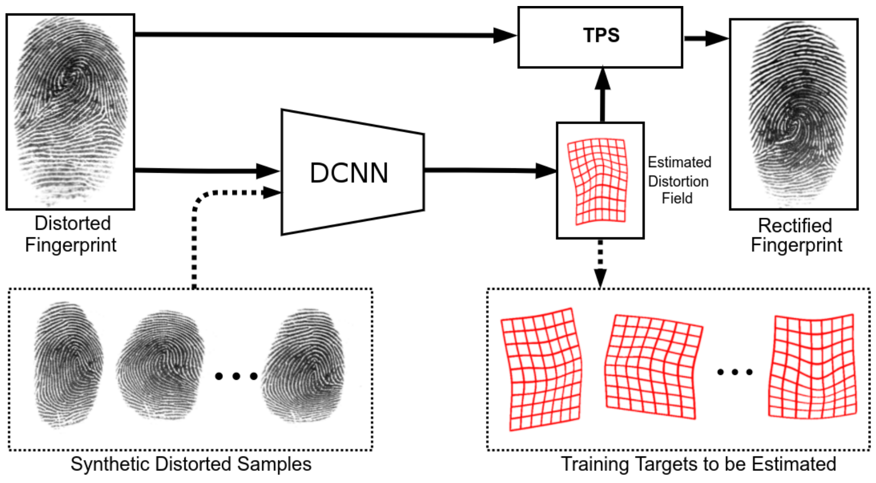

TPS
DCNN
Estimated Distortion Field
Distorted Fingerprint
Rectified Fingerprint
Synthetic Distorted Samples
Training Targets to be Estimated

Figure 8.1: Flowchart of the proposed method for rectifying distorted fingerprints. The solid line shows testing path and the dash line shows training path.

during recording procedure is a well-established approach. In this approach, the hardware detects distorted samples using different techniques, such as measuring excessive force [230] or the deformation of the acquisition surface [231], and motion processing during capturing fingerprint video [232]. The hardware rejects severely distorted records and asks the user to provide a new impression until the system requirements are satisfied. Despite the improvements in recognition performance [16], there are certain drawbacks associated with the use of hardware-based distortion detection techniques: (i) they need specific sensors and additional capabilities; (ii) it is not possible to apply them on previously recorded samples; (iii) it makes the system weak against malicious users who have altered their finger tips and ridge patterns; (iv) it is merely detecting distortion, and there is no rectification process since user is obligated to provide clean impressions.

Since geometric distortion essentially moves features in fingerprints, adding distortion tolerance to fingerprint matching has shown promising results in compensating for the distortion problem [233–238]. Distortion can be modeled by different special transformations such as rigid and thin plate spline (TPS) [239]. Although rigid transformation is not powerful enough to model the complex properties of geometric distortion, combining a global rigid transform and a local tolerant window have shown improvements in matching distorted samples [233, 234]. TPS as a more complex transformation has been used to make matching algorithms tolerant to geometric distortion [235]. However, compensating for distortion by adding tolerance to a fingerprint matcher inevitably results in a higher false positive match rate, and is highly dependent on estimating parameters of a complex transformation function.

Ross et al. [240,241] proposed a rectification technique based on learning deformation pattern from the correspondence of ridge curvatures of the same finger in different impressions. By computing average distortion based on corresponding ridges, it is possible to estimate parameters of the TPS transformation. This method showed improvement in matching distorted samples. However, the performance of the ridge curve correspondence method is highly dependent on the number of impressions of the same finger, and in most databases there are not enough samples per class to provide such an estimation.

Based on the assumption that the ridge frequency within a normal fingerprint is constant, Senior and Bolle [242] introduced a mathematical method of distortion rectification by equalizing the frequency map in distorted fingerprints. Their method improves matching performance, especially when applying equalization to both distorted and original samples before matching. Although it has been shown in [243, 244] that the ridge frequency map has discriminative information, and clearly it is not constant within the whole fingerprint area, their approach offered two important accomplishments compared to previous works. First, it does not need any specific hardware design, and second, it is possible to apply their algorithm on a single fingerprint image. However, equalizing all ridge spacings in a fingerprint has the following limitations: (i) some identification information will be lost and the false positive match rate will increase; (ii) in severe distortion cases, ridges are mixed together, and it is not possible to equalize the spacing between them; and (iii) equalizing the ridge frequency map within the whole fingerprint introduces distortion in the ridge orientation map.

More recently, Si et al. [8] collected the Tsinghua distorted fingerprint database by inducing 10 different types of force and torque to fingers during the fingerprint acquisition process. They proposed a statistical model for distortion by computing minutiae displacements in distorted and corresponding original samples. In this method, the top two significant principal components of displacement are used to generate a dictionary of distorted samples. For each input sample, the ridge frequency and orientation maps are computed and compared

Layer	**Type**	**Kernel Size**	**Input Size**	**Output Size**
1	Conv, BN, ReLU, MP	$3 \times 3 \times 32$	$256 \times 256 \times 1$	$128 \times 128 \times 32$
2	Conv, BN, ReLU, MP	$3 \times 3 \times 64$	$128 \times 128 \times 32$	$64 \times 64 \times 64$
3	Conv, BN, ReLU, MP	$3 \times 3 \times 64$	$64 \times 64 \times 64$	$32 \times 32 \times 64$
4	Conv, BN, ReLU, MP	$3 \times 3 \times 128$	$32 \times 32 \times 64$	$16 \times 16 \times 128$
5	Conv, BN, ReLU, MP	$3 \times 3 \times 256$	$16 \times 16 \times 128$	$8 \times 8 \times 256$
6	Conv, BN, ReLU, MP	$3 \times 3 \times 512$	$8 \times 8 \times 256$	$4 \times 4 \times 512$
7	Conv, BN, ReLU, MP	$3 \times 3 \times 1024$	$4 \times 4 \times 512$	$2 \times 2 \times 1024$
8	Conv, BN, ReLU, MP	$3 \times 3 \times 2048$	$2 \times 2 \times 1024$	$1 \times 1 \times 2048$
9	Conv	$1 \times 1 \times 2$	$1 \times 1 \times 2048$	$1 \times 1 \times 2$

Table 8.1: Architecture of the proposed DCNN used for estimating the distortion fields. All layers except the last one comprise Convolution (Conv), Batch Normalization (BN), ReLU and Max Pool (MP). All max poolings are 2×2 with the stride of two. All convolution strides are one, and all inputs to convolutions are padded to have the same size outputs.

to a dictionary in order to find the nearest distorted template. Their method shares all advantages of previous works, and it does not equalize the ridge frequency map. Therefore, discriminatory information of the frequency map is preserved and the ridge orientation map is not distorted. Considering all advantages of using a dictionary of distorted templates, there are still some limitations that need to be addressed: (i) computing frequency and orientation maps for input samples and comparing them with all samples in the dictionary takes a significant amount of time (from a second to several minutes depending on fingerprint properties); (ii) the performance of this method is related to the dictionary size, and increasing the dictionary size makes system slower; and iii) this method is highly dependent on computing the frequency and orientation maps of input samples which are not reliable due to the presence of distortion.

8.3 DCNN-based Distortion Estimation Model

Our method is inspired by the rectification approach proposed by Si et al. [8, 245]. The major limitation of their method is related to identifying the nearest distorted template in a dictionary of distorted samples. Finding the nearest neighbor to the distorted input sample in the dictionary is not accurate due to unreliable frequency and orientation maps extracted from the input sample. Instead of using a dictionary of the ridge frequency and orientation maps of distortion templates, we use a DCNN to estimate distortion parameter of the input sample. In this way, the non-linear transformations that caused distorted templates are being learned by the deep neural network during the training phase. The input to the network is the raw fingerprint image, and there is no need for computing the ridge frequency and orientation maps for the input samples. Contrary to the dictionary-based approach, the computational time of our proposed DCNN for estimating the distortion for an input, does not change by increasing the number of training samples since the network has a fixed number of parameters. On the other hand, the DCNN is capable of learning complex combinations of geometric distortions. A flowchart depicting the rectification scheme of the proposed method is shown in Figure 8.1. In the training phase, the network learns to estimate the distortion parameters of the input training images by minimizing the difference between the estimated parameters and the actual values. In the testing phase, the network estimates distortion parameters by mapping the input fingerprint to a non-linear manifold of distortion bases. Using the estimated distortion template and the input fingerprint, it is possible to rectify the distorted fingerprint by the inverse TPS [239] transformation of the distortion.

8.3.1 Modeling Geometric Distortion to Generate Synthetic Distorted Fingerprints

Training a DCNN requires a comprehensive database of labeled images. We generated a synthetic database of distorted images in order to train our network. It is essential to model distortion for this purpose. Similar to [8], we used the Tsinghua distorted fingerprint database to statistically model geometric distortion. To extract displacement due to geometric distortion, we matched minutiae pairs from the original and distorted fingerprint samples. Minutia detection was performed using VeriFinger 7.0 SDK [246]. Since minutiae are anomalies in the fingerprint ridge map and have random positions we defined a similar grid of points as in [8] to have a reference of distortion to be compared among different fingers. Using sampling grid pairs from the original and distorted fingerprints, it is possible to represent distortion as a displacement of corresponding points on the original grid and the distorted grid as follows:

$$\mathrm{d_i} = \mathrm{x_i^D} - \mathrm{x_i^N}, \tag{8.1}$$

where $\mathrm{d_i}$ is the displacement of minutia for the ith pair of distorted and the corresponding normal fingerprint. Using distortion samples of the Tsinghua database and computing the distortion fields, it is possible to statistically model distortion by its principal components using PCA [247–249]. Approximation of distortion fields using PCA will be:

$$\hat{\mathrm{d}} \approx \bar{\mathrm{d}} + \sum_{\mathrm{i}=1}^{\mathrm{t}} \mathrm{c_i} \sqrt{\lambda_{\mathrm{i}}} \mathrm{e_i}. \tag{8.2}$$

In the above equation, t is the number of selected principal components, $\mathrm{c_i}$ is the coefficient of the corresponding eigenvector component, $\mathrm{e_i}$ is ith eigenvector and λ_{i} is its corresponding eigenvalue. We used the first two significant eigenvectors of distortion to generate our synthetic samples. We generated a dataset of synthetic distorted fingerprints using 1033 normal fingerprints from the BioCOP 2013 dataset [250]. Each normal fingerprint was transformed to 400 distorted images by sampling each of the two principal distortion components extracted from the Tsinghua database. Sampling was performed randomly with a uniform distribution between -2 and 2. The generated dataset has $1033 \times 401 = 414,233$ samples, in which each ID has one normal sample and 400 distorted samples. Figure 8.2 shows two generated samples for two different fingers.

8.3.2 Network Architecture

We used a deep convolutional neural network to learn the two eigenvector-based distortion coefficients. Compared to the fully connected networks, DCNNs are more robust against over-fitting due to weight sharing and fewer learning parameters. All layers except the last one are convolutional layers. The input image to the network has a size of $256 \times 256 \times 1$

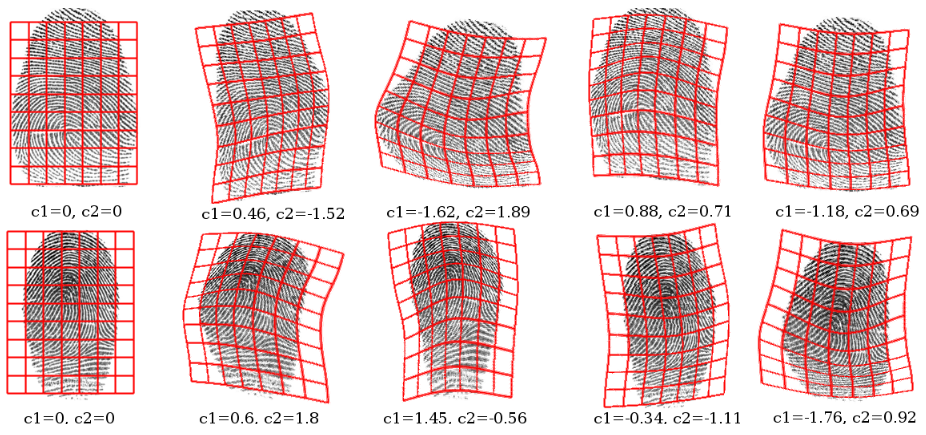


Figure 8.2: Examples of synthetic distorted fingerprint samples generated for training the network. Each sample is generated by randomly sampling distortion bases c_1, c_2.

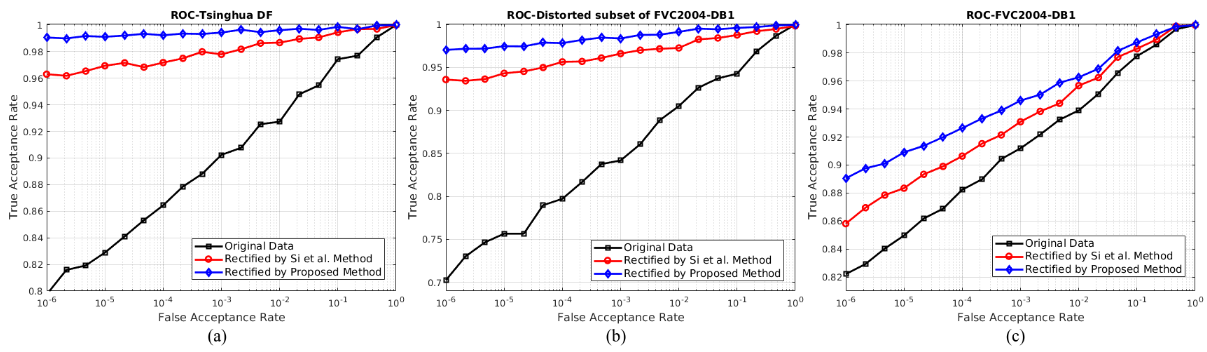


Figure 8.3: The ROC curves of three matching experiments for the following three databases (a) Tsinghua DF database, (b) FVC2004 DB1 and (c) geometrically distorted subset of FVC2004 DB1.

pixels (first dimension is width, second is height and third is the depth). Our network consists of 9 convolutional blocks. Each layer, except the last one, comprises convolution, batch normalization, Rectified Linear Unit (ReLU) and max polling with stride equal to two. A detailed properties of the network is shown in Table 8.1.

The network minimizes the norm-2 distance between ground truth coefficients (c_1 and c_2) and the DCNN outputs. For training the model, we first centered images according to the center of mass of the fingerprint area, and then scaled and cropped inputs to a size of 256×256. We used 401,000 synthetic distorted fingerprint images to train the model.

	Time (sec)	
Method	Tsinghua DF	FVC2004 DB1
Si et al. [8]	8.373	7.816
Our	0.741	0.736

Table 8.2: Average time of distortion estimation. The proposed DCNN distortion estimation method is approximately 10 times faster than the nearest neighbor method used by Si et al. [8].

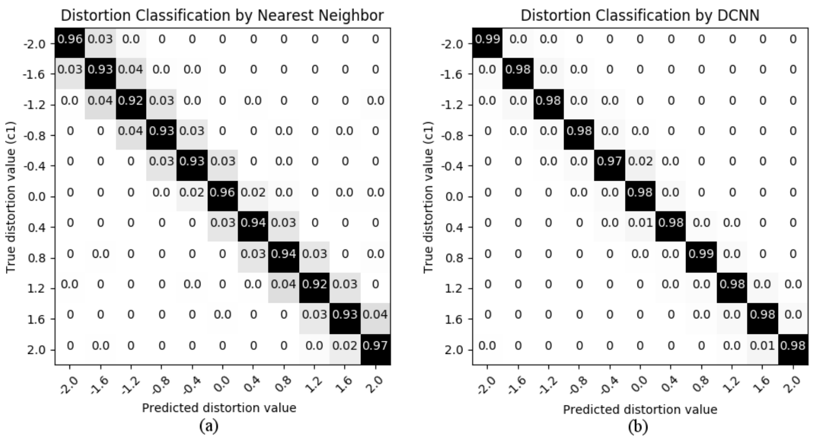


Figure 8.4: Confusion matrices for the following approaches (a) the nearest neighbor method by Si et al. [8] and (b) the proposed DCNN-based distortion estimation.

The network was trained over 40 epochs, each epoch consisting of 6,265 iterations with a batch size = 64. Adam optimization method [16, 251] is used as the optimizer due to its fast convergence with beta = 0.5 and learning rate = 10^{-4}.

8.4 Experiments

Our first performance measure for evaluating the proposed distortion rectification is the overall matching performance. To evaluate the contribution of the proposed method in improving matching performance, we conducted three experiments on each of the following three databases: FVC2004 DB1, distorted subset of FVC2004 DB1 and Tsinghua DF database. VeriFinger 7.0 SDK [246] is used to match fingerprint samples.

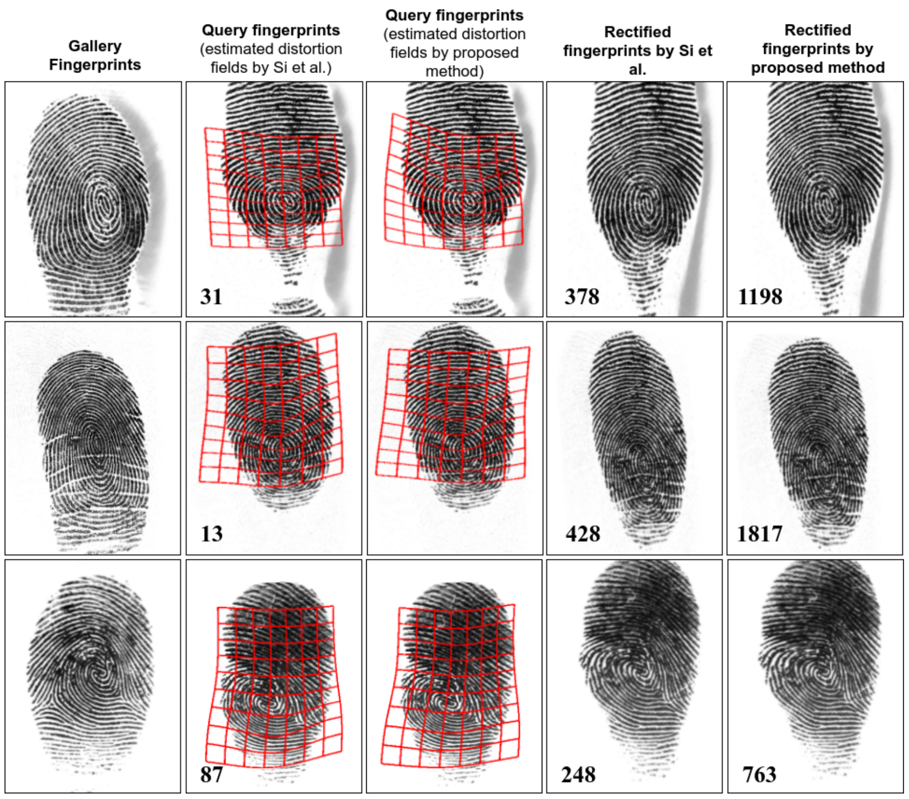

Gallery Fingerprints
Query fingerprints (estimated distortion fields by Si et al.)
Query fingerprints (estimated distortion fields by proposed method)
Rectified fingerprints by Si et al.
Rectified fingerprints by proposed method
31
378
1198
13
428
1817
87
248
763

Figure 8.5: Match scores for three pairs of normal and rectified fingerprints by two different approaches. The red grid on query fingerprints shows estimated distortion fields by our method and the method proposed by Si et al. [8]. Two first samples are from the Tsinghua DF database and the third sample is from FVC2004 DB1.

The match score in each experiment is calculated for pairs of samples with the same ID, and no imposter pairs are conducted since the match score of VeriFinger is linked to the false acceptance rate (FAR). Higher match scores have a lower chance of falsely being accepted. In all three matching experiments, the first sample in each pair is a normal fingerprint without distortion, and the second one is the original distorted sample or the rectified sample. Rectification is performed both by our method and the method proposed by Si et al. [8]. ROC curves on three databases are depicted in Figure 8.3.

In the first experiment, samples from the Tsinghua DF database are rectified to evaluate the training procedure of the network and the rectification performance. The Tsinghua DF database consists of 320 pairs of normal and distorted fingerprints from 185 different fingers.

Network training is performed using a synthetic distorted dataset generated by randomly sampling the first two significant principal components of the distortion manifold extracted from the Tsinghua DF database. Although the network has never seen the original samples from the Tsinghua DF database during the training procedure, distortion components used to generate the synthetic dataset may bias the performance of the network. Therefore, it is essential to evaluate matching performance on a dataset containing only geometric distortion that is different from the Tsinghua DF database. In the second experiment, a geometrically distorted subset of FVC2004 DB1 is used to evaluate the rectification performance of the proposed method. The distorted subset of FVC2004 DB1 contains 89 samples with skin distortions.

In the third experiment, FVC2004 DB1 is used to evaluate the rectification performance on a distorted database containing a variety of geometric and photometric distortions. FVC2004 DB1 consists of 110 classes and eight samples per class. Samples of each class are acquired by deliberately inducing photometric or geometric distortions. Since FVC2004 DB1 contains different distortion types, the proposed method targets only geometrically distorted samples and rejects other distortion types.

The quality of rectified distorted samples depends on the performance of the distortion estimation algorithm. We conducted an experiment to compare distortion estimation of DCNN with the nearest neighbor method used by Si et al. [8]. The synthetic distorted database used in this paper was generated using random sampling of the first two significant principal components. For comparison purposes, we generated another distorted database that was the same as Si et al. [8] to compare distortion classification of the two methods. The proposed DCNN estimates continuous values of distortion basis. Therefore, we quantized the network output to have 11 classes for each basis. In this order, class 1 is the first distortion basis with coefficient equal to -2.0, and class 11 is the first distortion basis with coefficient equal to 2.0. The confusion matrices for the two methods of classifying the first basis are shown in Figure 8.4. The Distribution of diagonal values of the second confusion matrix shows that the proposed DCNN is much more precise in estimating distortion coefficients. Although nearest neighbor is not accurate enough, it contributes to distortion rectification since it finds the target distortion class with an error margin of approximately two classes.

To compare the rectification results of our approach and the method proposed by Si et al. [8], three examples from the Tsinghua DF database and FVC2004 DB1 are shown in Figure 8.5. The rectified samples by both methods are very similar but the match score measurement indicates that there is a significant difference between them. A slight estimation error in distortion parameters prevents the spatial transformation from correctly restoring minutiae displacements.

In a fingerprint recognition system, distortion rectification is one of the preprocessing steps that can affect the total response time of the system. It is not possible nor efficient to use a computationally slow rectification method in a real-time recognition system since it brings inconvenience to users. Therefore, it is essential to evaluate the rectification speed. We conducted two experiments to evaluate the average response time of the rectification process on a PC with 3.3 GHz CPU and NIVDIA TITAN X GPU. Results are reported in Table 8.2. From the average response time of the proposed approach and the matching experiments, it can be observed that the proposed DCNN as a distortion estimator, not only increases the accuracy of distortion detection, but also significantly reduces the detection time.

An important fact to be considered is that the proposed algorithm is executed on the GPU, but the nearest neighbor method is executed on the CPU because it is not possible to implement a search method on parallel processors. Therefore, the reduction of the rectification time is mainly because of the capability of neural networks to embed training samples in the network parameters which enables us to convert a search problem to a direct prediction problem.

Additionally, contrary to the nearest neighbor method, the response time of the proposed DCNN is independent of the properties of input samples to the network, and guarantees an efficient lower bound for processing speed.

8.5 Conclusion

Geometric distortion significantly reduces the match score produced by a fingerprint verification system. In the positive recognition scenario, this causes inconvenience for users, but in the negative recognition scenario where users may intentionally distort their fingerprint, this can be considered as a security vulnerability. Therefore, it is essential to implement distortion rectification in order to prevent malicious users from hiding their identity, as well as reduce the inconvenience of using identification systems in authentication tasks. We proposed a novel approach to estimate distortion parameters from raw fingerprint images without computing the ridge frequency and orientation maps. A deep convolutional neural network is utilized to estimate distortion parameters of input samples. We successfully rectified distorted samples from the Tsinghua DF database and FVC2004 DB1 using the estimated distortion template. A comprehensive database of distorted samples was generated in order to train our deep neural network. The experimental results on several databases showed that the DCNN can estimate the non-linear distortions of samples more accurately. Comparing to the previous works, our method decreased rectification time significantly by embedding the training samples in the network parameters. In addition, since the estimation time of the proposed method is independent of the training size, it is possible to increase the number of principal components which are used to generate the synthetic distorted database for the future works.

Chapter 9

Latent Fingerprint Reconstruction Using GANs

9.1 Introduction

Automatic fingerprint recognition systems have been widely adopted to perform reliable and highly accurate biometric identification. Compared to other biometric traits, such as iris, the fingerprint has a unique superiority of being collected indirectly from crime scenes from latent friction ridge impressions. Fingerprint samples can be categorized into three main groups based on the acquisition techniques such as: inked, live-scan, or latent samples. The inked and live-scan samples are considered as clean samples for which users leave impressions intentionally in access control or authentication scenarios. In addition, an agent, or the acquisition process of the system itself, can monitor quality of the samples, and guide users to leave appropriate fingerprints. In past decades, algorithms for preprocessing and matching clean fingerprints have advanced rapidly, resulting in the development of numerous and varied commercial fingerprint recognition systems.

In contrast, latent fingerprints are the marks of fingers unintentionally left on the surface of an object in a crime scene. Typically, a latent fingerprint is a 'noisy' image with a notable missing area, therefore containing a lower amount of ridge information (i.e. minutiae) compared to inked or live-scan fingerprints. Processing latent fingerprints is a complex and challenging problem due to the under-determined properties of the problem, and presence

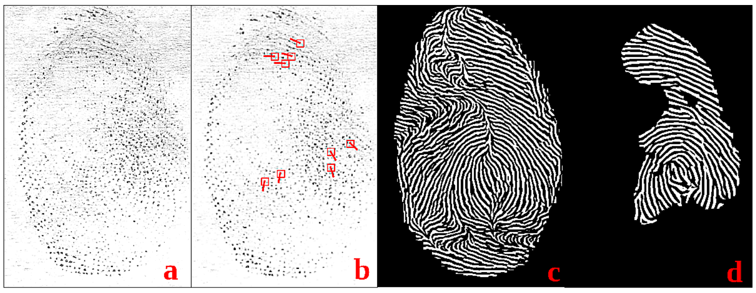


Figure 9.1: Examples of different latent fingerprint reconstruction methods: a) a latent fingerprint with severe distortion and missing area, b) minutiae-based prediction using [15], c) ridge-based reconstruction using [16], d) constrained ridge-based reconstruction using the proposed algorithm.

of many disturbing factors introduced by background objects or patterns on the substrate or surface, the force and torque involved in depositing the latent print, etc. Commercial and state-of-the-art methods for recognizing the inked or live-scan fingerprints often fail to process latent samples, even in the preprocessing stage [252]. Therefore, various approaches have been proposed in the literature to tackle the problem of latent [253–257] and distorted [245, 258, 259] fingerprints.

Enhancing latent fingerprints often leads to optimizing a cost function that measures the quality of reconstruction by comparing reconstructed information and their ground truths. Based on the type of the reconstructed information, latent fingerprint reconstruction methods can be divided into two main categories: ridge-base and minutiae-based methods.

In the ridge-base methods [16, 221, 254–256], algorithms try to predict the ridge information, which can be the orientation map or the ridge pattern itself, and minimize the similarity between the generated information and their ground truths. Then minutiae information can be extracted from the predicted ridge information. These methods are optimized to predict the ridge information without estimating the local quality of the input samples. For parts of the input latent fingerprint that there exists some information, typically these algorithms produce useful results. However, for the parts in which there is a severe distortion, not only these algorithms can not predict the missing information, but also, they can destroy the ID information by generating erroneous minutiae. Figure 9.1(c) shows a reconstructed ridge map for a latent fingerprint with severe distortion. As the reconstructed ridge map indicates, algorithm [16] generates meaningful ridge information for parts that contain some ridge information in the input latent sample, but for other parts it produces random ridge patterns which drastically decreases the matching score.

On the other hand, minutiae-based methods [15, 260, 261] directly predict the type and location of minutiae of the latent fingerprints without reconstructing the ridge pattern. Minutiae-based reconstruction methods are more robust against severe distortion or missing areas of the input latent fingerprint, since they predict the probability of a minutia by analyzing a small area around each candidate ridge point, and they reject large missing areas. Hence, this rejection drastically decreases the number of founded minutiae. Figure 9.1(b) shows some minutiae that were detected using local processing of a latent fingerprint.

In case of severely distorted latent fingerprints, both ridge-based and minutiae-based reconstruction methods fail. Ridge-based reconstruction methods fail because they fill missing areas with incorrect ridge patterns, therefore they introduce minutiae which change the ID of reconstructed samples. Minutiae-based prediction methods fail because they reject most of the missing areas, and at the end, they often predict fewer minutiae, which are not enough to identify samples.

Recently, machine learning, especially deep learning, has demonstrated significant performance in many fields including biometrics [48, 262–275]. In this study we developed a deep convolutional neural network (DCNN) model to reconstruct the ridge information of latent fingerprints. The core network in the model is a conditional generative adversarial network (cGAN) that reconstructs the obscured ridge information of the latent samples. To overcome the limitation of the previous ridge-based reconstruction methods, our model predicts three extra maps in addition to the ridge map: the orientation, frequency and segmentation maps. Generating the orientation and frequency maps ensure that the model is considering the orientation and frequency information of the input latent fingerprints. Generating a segmentation map prevents the model from filling large missing areas in the input latent samples; thus, it optimizes the amount of ridge information that can be reconstructed. In addition, to force the generator to preserve the ID information (type and location of minutiae), we developed an auxiliary deep model to extract the perceptual ID information (PIDI)

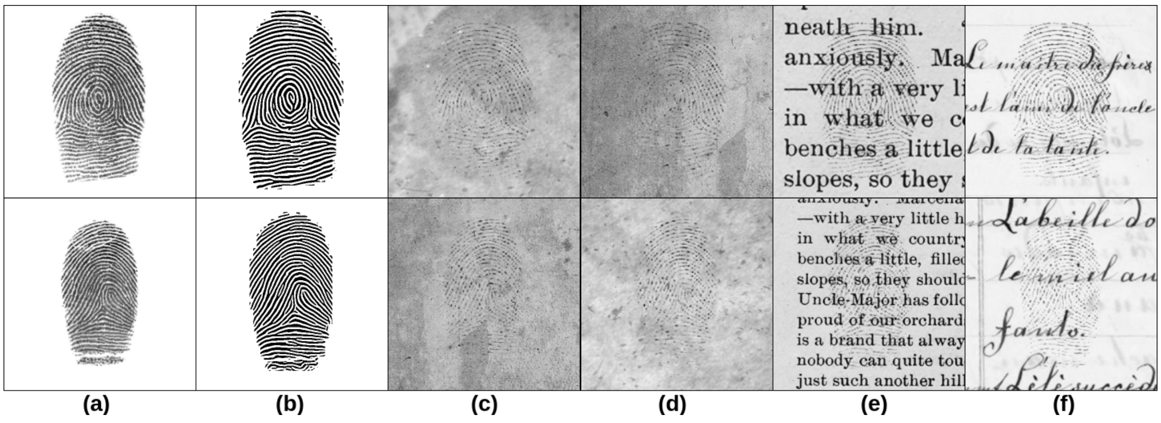


Figure 9.2: Examples of synthetically generated latent fingerprints: (a) is the original fingerprint, (b) is the corresponding binary ridge map, and (c, d, e, f) are generated latent fingerprints.

of the generated sample and fuse it into the cGAN model to enhance the reconstruction process.

9.2 Method

9.2.1 Conditional Generative Adversarial Networks (cGANs)

GANs [27] are one of the most popular groups of generative networks. Generative networks map a sample from a random distribution $\mathrm{p_z(z)}$ to a target domain of desired samples $\mathrm{y = G(z, \boldsymbol{\theta}_g) : z \rightarrow y}$, through training parameters $(\boldsymbol{\theta}_\mathrm{g})$ of the network. GANs are different from conventional generative models because they prosper from a discriminator network. The discriminator network compares the generated samples (fake samples) $\mathrm{y = G(z, \boldsymbol{\theta}_g)}$ with the real samples from the target domain, and tries to distinguish between them. Simultaneously, the generator (typically an auto-encoder) tries to fool the discriminator by generating more realistic samples. In each iteration, the generator produces better samples in an attempt to fool the discriminator, and the discriminator improves by comparing the real samples with the generated samples. In other words, the discriminator D and the generator G play a

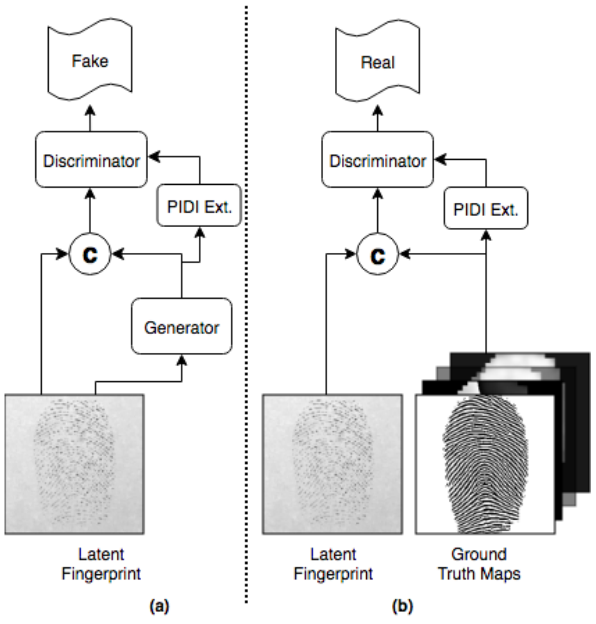


Figure 9.3: Training criteria of the discriminator. The discriminator receives two types of inputs, (a) the generated fingerprint maps, and (b) the ground truth maps. In both scenarios, the corresponding latent fingerprint is concatenated (**C**) to either the generated or the ground truth maps to act as the condition. The auxiliary verifier module extracts the PIDI from both generated maps and ground truths, and passes them to the discriminator. The discriminator learns to distinguish between the real maps and fake maps based on the quality and PIDI of generated maps.

two-player minimax game with the following objective function:

$$\mathrm{V_{GAN}(G,D) = E_{y\sim P_{data}(y)}[logD(y)] + E_{z\sim P_z(z)}[log(1-D(G(z)))]}, \tag{9.1}$$

where generator G tries to minimize Equation 9.1 and discriminator D tries to maximize it. An additional L2 or L1 distance loss is added to the objective function in the literature to force the network to generate samples which are closer to the target ground truth. The final generator model trains as follows:

$$\mathrm{G_{optimal}} = \min_{\mathrm{G}} \max_{\mathrm{D}} \mathrm{V_{GAN}(G,D)} + \lambda \mathrm{l_{L1}(y^*, G)}, \tag{9.2}$$

where λ is the coefficient of L1 distance and $\mathrm{l_{L1}(y^*, G)}$ is:

$$\mathrm{l_{L1}(G) = ||y^* - G(z)||_1}, \tag{9.3}$$

where $\mathrm{y^*}$ is the ground truth for the output of the generator.

In the real-world application of restoring a latent fingerprint image, both the source and target domains are available for training. Therefore, Isola et al. [276] proposed the conditional GAN model which has two modifications compared to conventional GANs. First, instead of using noise as the input to the network, real training samples from the source domain are fed to the network $\mathrm{y = G(x, \theta_g) : x \to y}$. Second, they put a condition on the discriminator by concatenating the input sample with the generated sample. Therefore, the discriminator judges a generated sample based (conditioned) on the original sample which was fed to the network. The new objective function for the cGAN is:

$$\mathrm{V_{cGAN}(G,D) = E_{x\sim P_{data}}[logD(x,y)] + E_{x\sim P_{data}}[log(1-D(x,G(x)))]}. \tag{9.4}$$

9.2.2 cGAN for Latent Fingerprint Reconstruction

The formulation and the network architecture of cGAN proposed by Isola et al. [276] is a universal setup for image-to-image translation. However, processing a biometric image such as latent fingerprints is often more complicated than other image types due to the identification information that is embedded in the pattern of the image which should be preserved during the reconstruction process. For this purpose, we perform two modifications to the formulation and the network architecture to emphasize the ID of the input sample.

First, to increase the quality of reconstruction, we force the network to generate four fingerprint maps $Y = G(x, \theta_g)$ for each latent fingerprint input as follows:

$$Y = [y^R, y^F, y^O, y^S], \tag{9.5}$$

where, Y is the new output of the generator, and consists of the ridge map (y^R), the orientation map (y^O), the frequency map (y^F) and the ridge segmentation map (y^S) of the input latent fingerprint. These four maps are concatenated depth-wise to form Y. Figure 9.4(b) demonstrates the new output of the generator.

Equation 9.3 computes the direct reconstruction error. For the new outputs of the generator, the same formula is extended as:

$$l_{L1}(G) = \alpha_R||y^{*R} - y^R||_1 + \alpha_F||y^{*F} - y^F||_1 + \alpha_O||y^{*O} - y^O||_1 + \alpha_S||y^{*S} - y^S||_1, \tag{9.6}$$

where, y^{*R}, y^{*F}, y^{*O} and y^{*S} are the ground truth values for the ridge, frequency, orientation and segmentation map respectively, and α_R, α_F, α_O and α_S are weights for scaling loss of reconstruction of each generated map. Since all maps except the ridge map have low-pass characteristic and may prevent the generator from predicting the ridge map with sufficient detail, we set α_R to 1.0 and all other coefficients (α_F, α_O and α_S) to 0.1.

Second, the generator should preserve the identification information embedded in minutiae and ridge patterns. To extract and preserve the identification information we developed a method that is inspired by perceptual loss [51, 277–279] and multi-level feature abstraction [280–283]. For this purpose, we separately trained a deep Siamese CNN as a fingerprint verifier. The trained model is used to extract the perceptual ID information (PIDI) of the generated maps. Extracted PIDI are output feature maps of the first four convolutional layers of the verifier module, and are concatenated to the corresponding layers of the discriminator to emphasize the ID information on the discriminator's decision. Figure 9.4(d) shows how the output feature maps of the verifier contributes to the discriminator's decision making. Figure 9.3 shows the training criteria of the discriminator in more detail.

Table 9.1: Architecture of the PIDI extractor. All layers except the last one comprise Convolution (C), Batch Normalization (B) [9] and ReLU (R). The output of the last layer is flatted after the convolution. Convolution strides of all layers are two, and the spacial size of all kernels is 4×4.

L #	Type	K. Size	Input Size	Output Size
1	C, B, R	64	256×256×4	128×128×64
2	C, B, R	128	128×128×64	64×64×128
3	C, B, R	256	64×64×128	32×32×256
4	C, B, R	512	32×32×256	16×16×512
5	C, B, R	512	16×16×512	8×8×512
6	C, B, R	512	8×8×512	4×4×512
7	C	512	4×4×512	2048×1

9.2.3 Network Architecture

The proposed model consists of three networks: a fingerprint PIDI extractor, a generator, and a discriminator. The fingerprint PIDI extractor is one tower from a deep Siamese [284] fingerprint verifier that is trained using a contrastive loss [284]. We fix all weights of the CNN tower, and use it to extract the PIDI of samples generated by the generator. Figure 9.4(d) shows how the output of the first four layers of the PIDI extractor are fused into the discriminator to provide the identification information for the discriminator. Table 9.1 details the architecture of the PIDI extractor.

The generator is a 'U-net' [23, 276] auto-encoder CNN. In the 'U-net' architecture, some layers from the encoder are concatenated to layers of the decoder to keep the high frequency details of the input samples, and increase the quality of reconstruction in the decoder. Figure 9.4(a) shows a diagram of the generator, and Table 9.3 details the structure of the generator.

The discriminator is a deep CNN which maps the conditioned output of the generator with a size of $256 \times 256 \times 5$ to a discrimination matrix of size $16 \times 16 \times 1$. To force the discriminator to consider the ID information of the input samples, outputs of the first four layers of the PIDI extractor network are concatenated to the output of the corresponding layers of the discriminator. Table 9.2 details the architecture of the discriminator network. The discriminator judges the output of the generator in a patch-wise manner. Each element of the discriminator's output map represents the discriminator's decision about a corresponding patch in the generator's output.

The whole generative model was trained over 400 epochs, each epoch consisting of 7,812 iterations with a batch size = 64. Adam optimization method [251] is used as the optimizer due to its fast convergence with beta = 0.5 and learning rate = 10^{-4} .

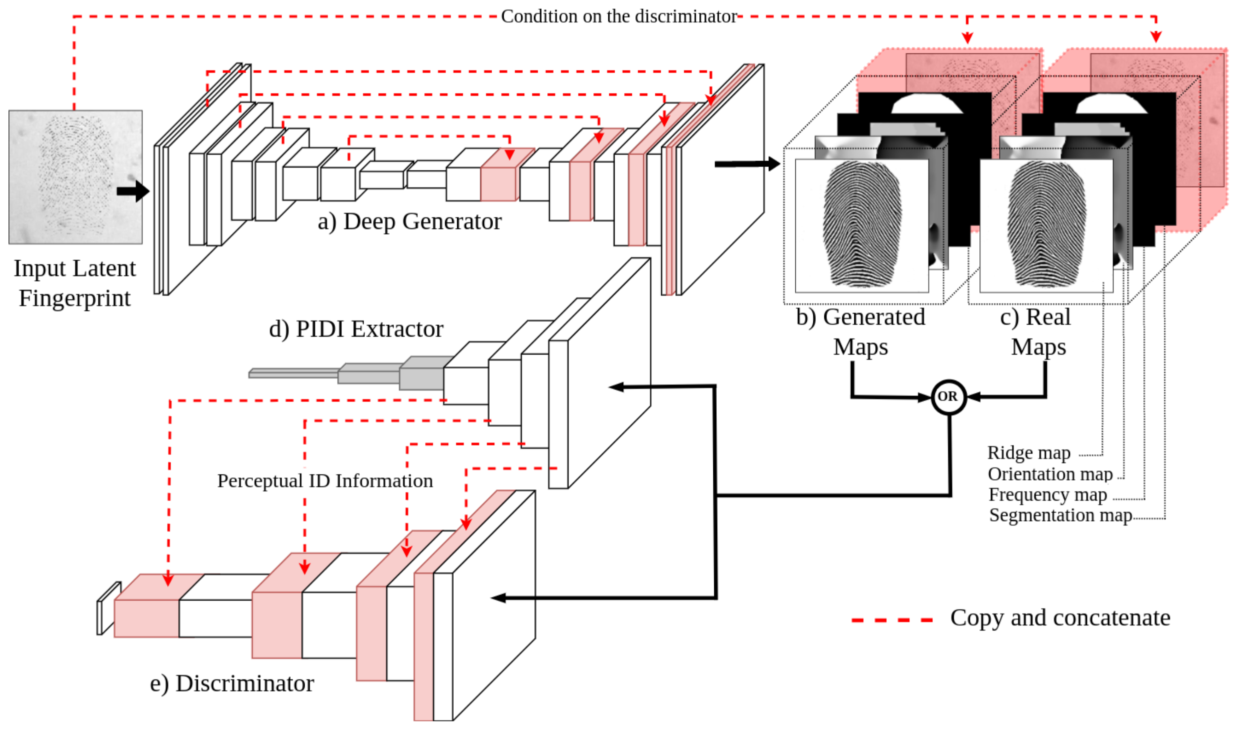


Figure 9.4: Complete diagram of the model. a) The deep generator takes the input latent fingerprints and generates a ridge, frequency, orientation and segmentation map simultaneously. b) Generated maps are concatenated with the input latent fingerprint to provide a condition for the discriminator. c) Real maps are extracted from the original clean fingerprints that were distorted to provide synthetic latent samples. During the training phase these maps are used to provide the supervision for the discriminator. d)One tower from a deep Siamese fingerprint verifier that was trained separately takes the generated or real maps and provides PIDI for the discriminator. e) The discriminator tries to distinguish between generated maps and the real maps using the combined ID and quality information from the generated maps.

Table 9.2: Architecture of the discriminator. All layers except the last one comprise Convolution (C), Batch Normalization (B) [9] and ReLU (R). Non-linear function of the last layer is a Sigmoid (S). Convolution strides of the first four layers are two. The output of the first four layers of the verifier are concatenated to the corresponding four layers of the discriminator to enforce ID information to discriminator's decision making.

Layer	Type	St.	Kernel Size	Input Size	Output Size	Concat. with	Final Output Size
1	C, B, R	2	$4 \times 4 \times 64$	$256 \times 256 \times 5$	$128 \times 128 \times 64$	L 1	$128 \times 128 \times 128$
2	C, B, R	2	$4 \times 4 \times 128$	$128 \times 128 \times 128$	$64 \times 64 \times 128$	L 2	$64 \times 64 \times 256$
3	C, B, R	2	$4 \times 4 \times 256$	$64 \times 64 \times 256$	$32 \times 32 \times 256$	L 3	$32 \times 32 \times 512$
4	C, B, R	2	$4 \times 4 \times 512$	$32 \times 32 \times 512$	$16 \times 16 \times 512$	L 4	$16 \times 16 \times 1024$
5	C, B, S	1	$4 \times 4 \times 1$	$16 \times 16 \times 1024$	$16 \times 16 \times 1$	-	$16 \times 16 \times 1$

Table 9.3: Architecture of the generator. All layers except the last one comprise Convolution (C), Batch Normalization (B) [9] and ReLU (R). Non-linear function of the last layer is Sigmoid (S). Convolution strides of the first four layers are two, and the spacial size of all kernels is 4×4.

#L	T	S	#K	Input Size	Output Size	#L	T	S	#K	Input Size	Output Size	Cn
1	C	1	64	256×256×1	256×256×64	10	C	1	512	16×16×512	16×16×512	-
2	C	2	128	256×256×64	128×128×128	11	D	2	512	16×16×512	32×32×512	L7
3	C	1	128	128×128×128	128×128×128	12	C	1	512	32×32×1024	32×32×512	-
4	C	2	256	128×128×128	64×64×256	13	D	2	256	32×32×512	64×64×256	L5
5	C	1	256	64×64×256	64×64×256	14	C	1	256	64×64×512	64×64×256	-
6	C	2	512	64×64×256	32×32×512	15	D	2	128	64×64×256	128×128×128	L3
7	C	1	512	32×32×512	32×32×512	16	C	1	128	128×128×256	128×128×128	-
8	C	2	512	32×32×512	16×16×512	17	D	2	64	128×128×128	256×256×64	L1
9	C	1	512	16×16×512	16×16×512	18	C	1	4	256×256×128	256×256×4	-

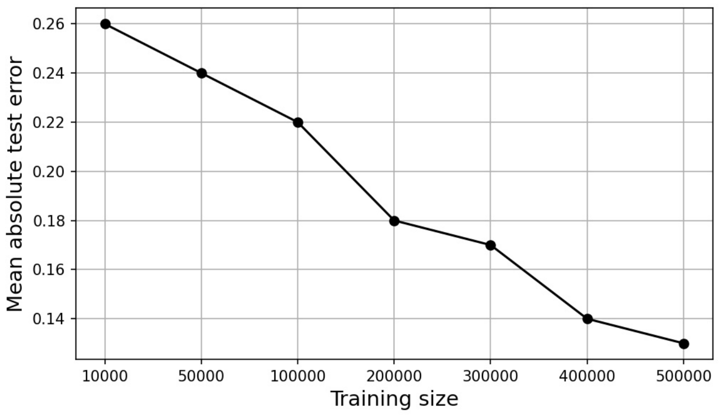


Figure 9.5: Evaluating the relation between the training performance, *i.e.* reconstruction error, and the size of the training set.

9.2.4 Synthetic Datasets of Latent Fingerprints

Training a generative model requires a large number of training samples. We synthetically generated databases of latent fingerprints by distorting clean fingerprints from the BioCOP 2012 and 2013 [250] databases. Initially, we distorted 5,000 fingerprints and for each fingerprint 100 distorted versions are generated. This dataset consists of 500,000 latent fingerprints and their corresponding clean samples. Examples of generated latent samples are shown in Figure 9.2. During our evaluations, we observed that the size of the training set significantly affects the reconstruction performance. Hence, we conducted an experiment to verify this hypothesis. We trained a cGAN model on training sets with different sizes of $\{10000, 50000, 100000, 200000, 300000, 400000, 500000\}$ and measured the test error. Figure 9.5 presents the results for the evaluation. We clearly observe that the performance of reconstruction is highly dependent on the size of the training set. However, unlike many other domains that have abundant data for the training, in biometric identification especially latent fingerprint reconstruction, the data is scars and the annotation is laborious and expensive. Hence, in the next Section, we consider several other approaches to address the scarcity of the data in latent fingerprint reconstruction tasks.

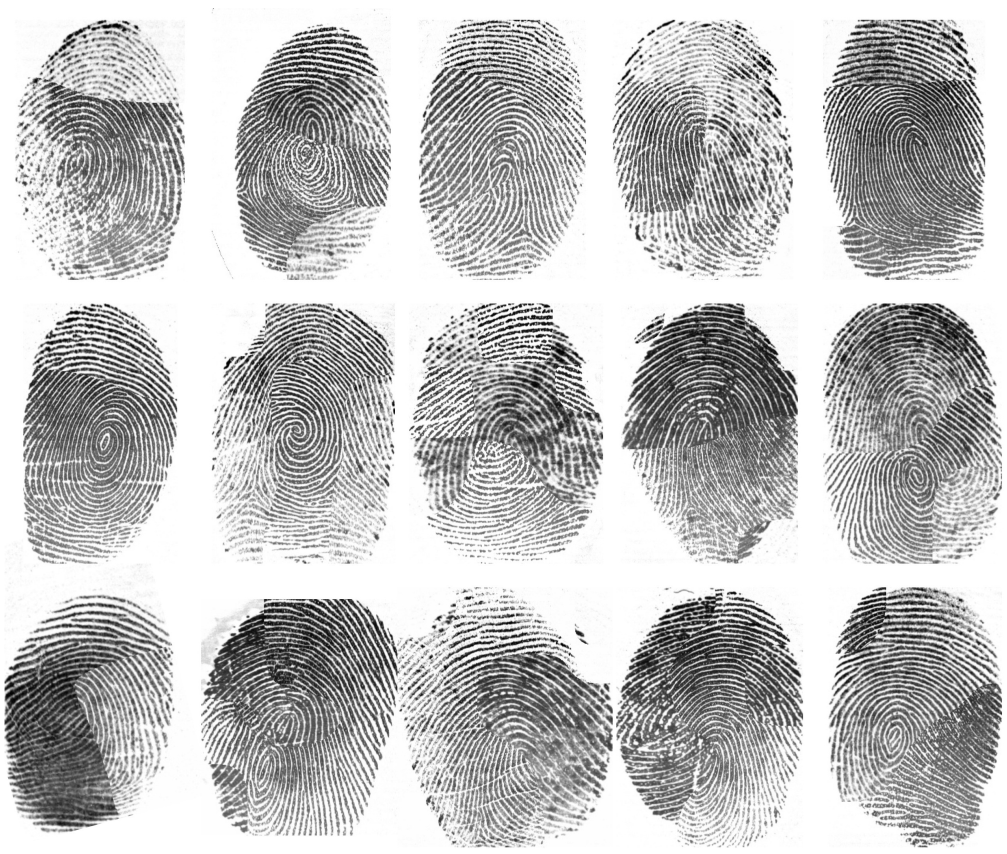


Figure 9.6: Examples of mixed fingerprints generated using our method. Each mixed image is obtained by combing two clean fingerprints from the dataset.

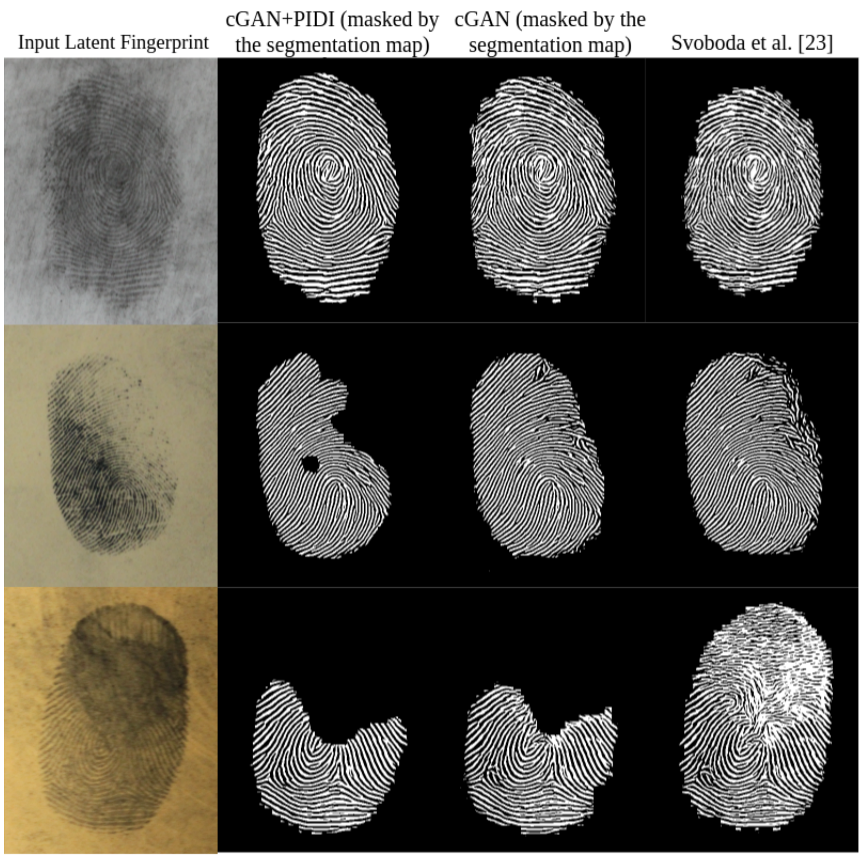

Input Latent Fingerprint
cGAN+PIDI (masked by the segmentation map)
cGAN (masked by the segmentation map)
Svoboda et al. [23]

Figure 9.7: Examples of the fingerprint reconstructions on real latent fingerprints. For each fingerprint image, the corresponding latent sample and the reconstructed sample are demonstrated. Matching scores for the latent fingerprints and the reconstructed samples are calculated using VeriFinger. All samples are from the IIIT-Delhi latent fingerprint database [17].

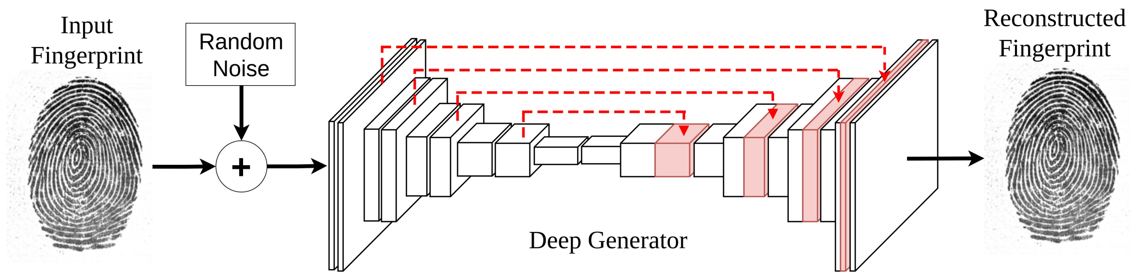


Figure 9.8: Diagram of denoising autoencoder. The UNet autoencoder takes a noisy version of the input fingerprint and tries to reconstruct the original input without noise.

9.2.5 Addressing Scarcity of the Data

To address the scarcity of the data, we develop two approaches. In the first approach, we constructed a second synthetic training set. We increased the number of clean source fingerprints to 20,000 samples and reduced the number of random distortions to 50. In this way, we enhanced the diversity of the dataset which has a direct impact on the generalization of the training model. Furthermore, we designed a supervised mixing augmentation technique presented in Section 7 which augments the dataset by combining images in the given dataset. Supervision for the mixing is obtained from the deep verifier network. In this way, we increased the number of clean fingerprints which are used to generate the synthetic latent fingerprints. Figure 9.6 provides several examples of our mixed fingerprints. The policy for distorting fingerprints in this dataset consists of random combination of geometric (additive noise, blurring, information drop, text overlay, dirt overlay, local contrast alteration, and color manipulation) and photometric distortions (affine transformations, non-affine transformation mimicking the plasticity of finger tips) with random magnitudes. The final database has one million distorted fingerprints and the corresponding ground truth fingerprints.

In the second approach, we used unsupervised pertaining to learn from the clean fingerprints without any supervision, and then, we transferred the learned knowledge to a new model for training on latent fingerprints. To this aim, we developed two models. In the first model, the generator is trained as a denoising autoencoder, *i.e.*it takes noisy version of the input fingerprints and tries to reconstruct the clean fingerprint without the noise. Figure 9.8 illustrates the diagram for this model. In the second model, we use contrastive learning to train the encoder part of the generator. For this purpose, we build upon SimCLR [285] approach in which the network tries to maximize the agreement between the representations belonging to different views of the same input while minimizing the agreement of the representations between two different inputs. Figure 9.9 presents the diagram for training this model. After pre-training the model using either of these methods, we transfer their knowledge to our target task, *i.e.*latent fingerprint reconstruction, by using the trained model as the generator in our main model.

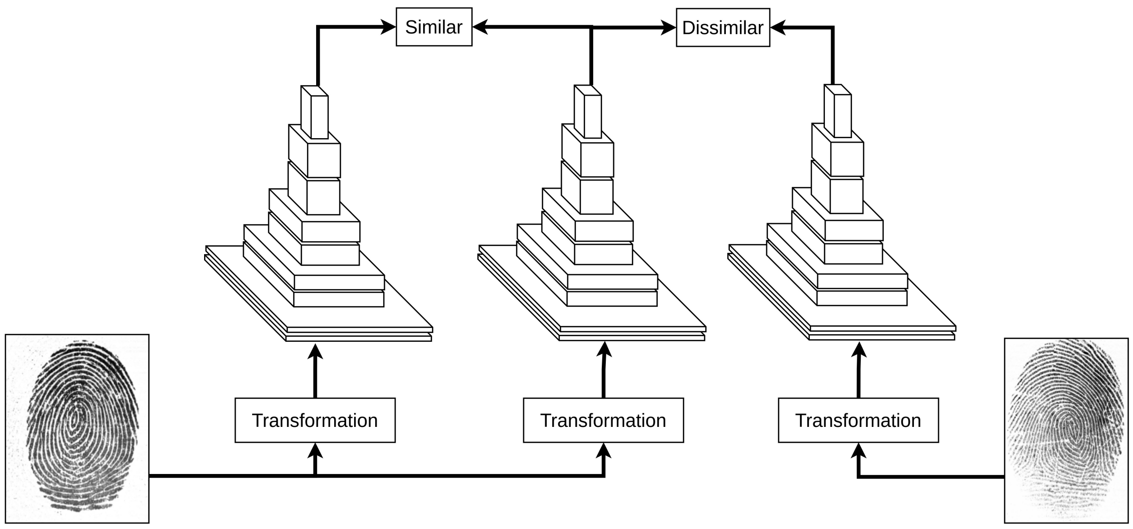


Figure 9.9: Diagram of unsupervised pre-training using contrastive learning. The encoder network learns to generate similar representations when the inputs are from the same fingerprint and generate dissimilar representations when the inputs are from different fingerprints.

9.3 Experiments

To evaluate the performance of the proposed latent fingerprint reconstruction technique, we conducted three different experiments on publicly available datasets of latent fingerprints. Unfortunately, NIST-SD27 [286] is no longer available, so the IIIT-Delhi Latent fingerprint [287] and IIIT-Delhi Multi Sensor Latent Fingerprint (MOLF) [288] databases were used to evaluate the proposed method. In all experiments, we used VeriFinger 7.0 SDK [246] and the NIST Biometrics Image Software (NBIS) [289] to match the reconstructed samples. In addition, to evaluate the role of the PIDI fusion which is developed to force the generator to preserve the ID information, we developed a second model which is exactly the same as our complete model but without the PIDI extractor module. Results for the complete model are named 'cGAN+PIDI', and results for the second model are named 'cGAN'.

9.3.1 Latent-to-sensor matching

For the first experiment, we used the IIIT-Delhi MOLF database. This database contains 19,200 fingerprint samples from 1000 classes (10 fingers of 100 individuals). For each ID, a

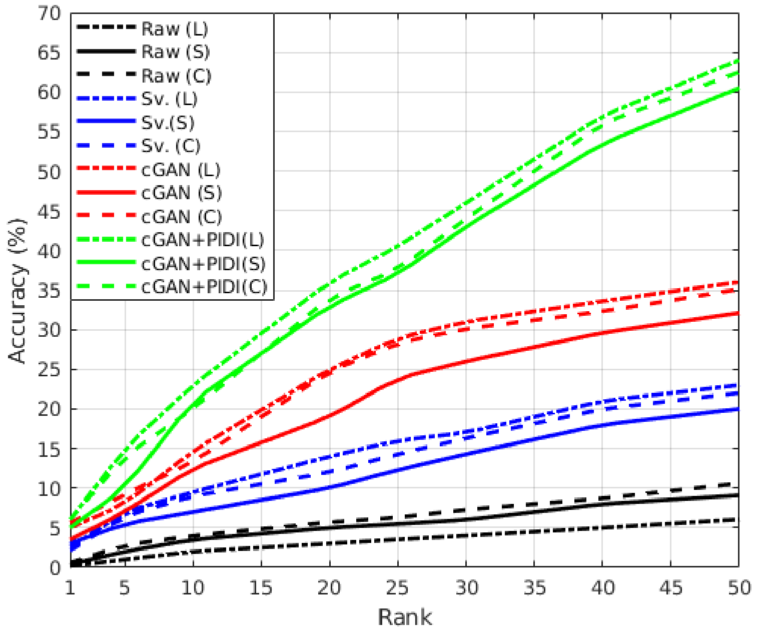


Figure 9.10: CMC curves for the experiment of latent-to-sensor matching with NBIS. The reconstructed ridge maps were matched to fingerprints captured by the Lumidigm (L), Secugen (S) and Crossmatch (C) sensors.

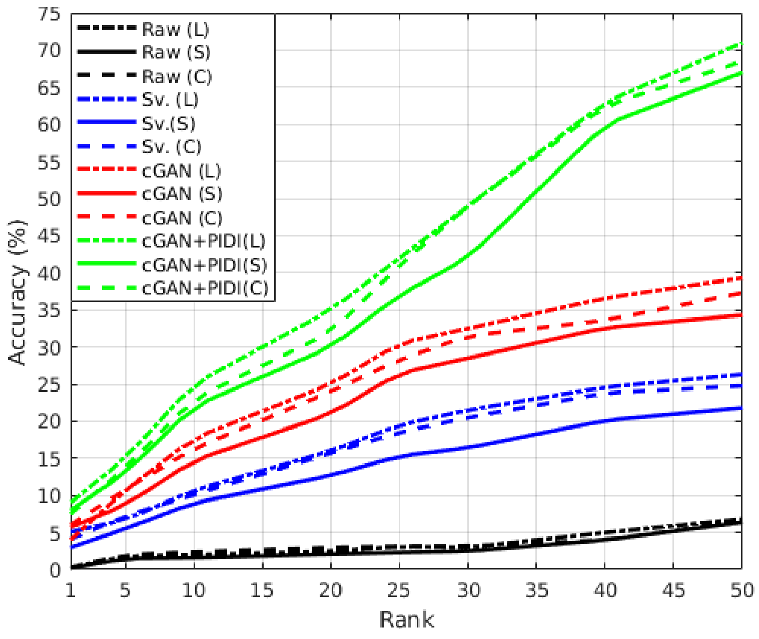


Figure 9.11: CMC curves for the experiment of latent-to-sensor matching with VeriFinger. The reconstructed ridge maps were matched to fingerprints captured by the Lumidigm (L), Secugen (S) and Crossmatch (C) sensors.

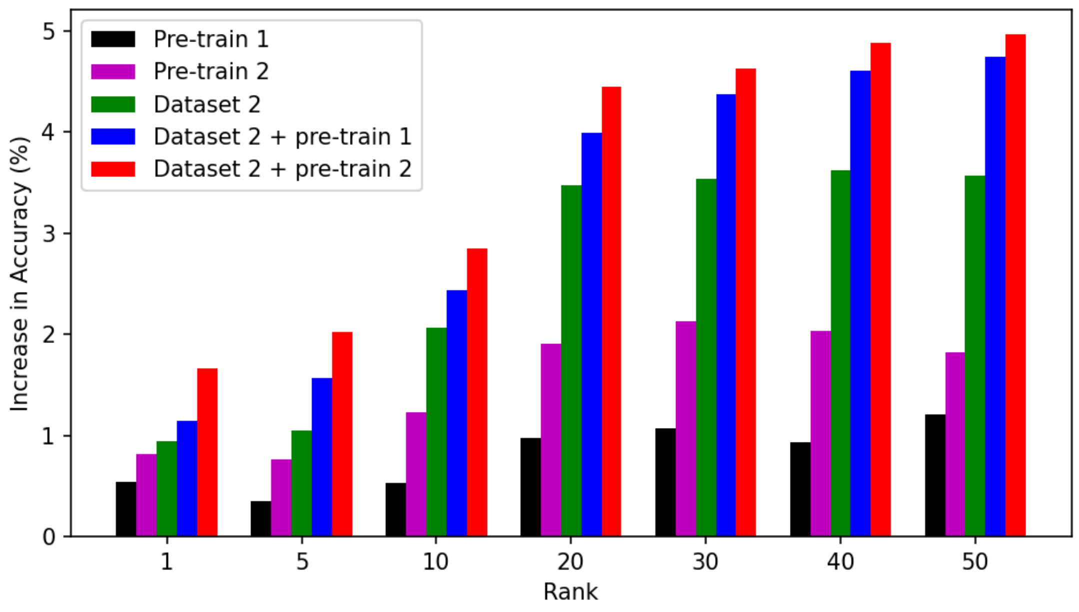


Figure 9.12: Improvement in CMC performance obtained using VeriFinger. Each bar shows the increase in the average performance over three sensors in the latent-to-sensor matching experiment.

set of latent fingerprint samples and the corresponding clean samples captured from three different commercial fingerprint scanners (Crossmatch, Secugen, Lumidigm) are available. As in the testing protocol established by Sankaran et al. [288], the first and second fingerprint samples of each user captured by a sensor are selected as the gallery. The entire latent fingerprint database consisting of 4,400 samples used as the probe set. CMC curves for this experiment are shown in Figure 9.10 and Figure 9.11. Table 9.4 and Table 9.5 show rank-25 and rank-50 accuracy for the latent-to-sensor matching experiment.

9.3.2 Latent-to-latent matching

For the second experiment, we evaluated the proposed method on the IIIT-Delhi latent fingerprint dataset which contains 1046 samples from all ten fingerprints recorded from 15 subjects. The experimental setup is defined the same as [287] by randomly choosing 395 images as gallery and 520 samples as probes. The CMC curves for this experiment is shown in Figure 9.14. Rank-1, rank-10 and rank-25 results are also shown in Table 9.6.

9.3.3 Improvement via Augmentation and Pre-training

In this part, we evaluate the improvement in the performance of the model obtained by utilizing approaches in Section 9.2.5 that aim to address the scarcity of annotated data in reconstructing latent fingerprints. This part consists of 5 experiments. In the first two experiments, we initialize our best model (cGAN+PIDI) using the weights obtained by pre-training approaches of denoising autoencoder and unsupervised contrastive learning, referred to as the 'pre-train 1' and 'pre-train 2', respectively. Then we finetune the model on our first synthetic dataset. In the third experiment, we train our cGAN+PIDI on the second synthetic dataset which is constructed using the modified protocol combined with supervised mixing augmentation, *i.e.*, the dataset with over 1 million images. For the last two experiment, we finetune our pre-tained models (experiments 2 and 3) using the second dataset. Afterward, we reconstruct the missing ridge information using each of the models and compare their matching performance in the latent-to-sensor paradigm using VeriFinger. The final performance is obtained by averaging the matching performance over the three sensors. To facilitate the interpretation, we plot the improvement obtained in each experiment compared to the our best performing model presented in the previous sections. Figure 9.12 illustrates the results for these evaluations.

We observe that both 'pre-train 1' and 'pre-train 2' improves the performance of reconstruction and result in an average improvement of 0.8% and 1.5%, respectively. This suggests that unsupervised contrastive learning provide superior performance for the pre-training task. Furthermore, we observe that using the second dataset which consists of more than a million images constructed using supervised mixing provides notable improvement over the same model trained on the first dataset. This further validates that our supervised mixing augmentation is a powerful tool for augmenting the latent fingerprint datasets. In addition, we observe that the combination of the pre-training with the extended dataset further improves the performance of the reconstruction.

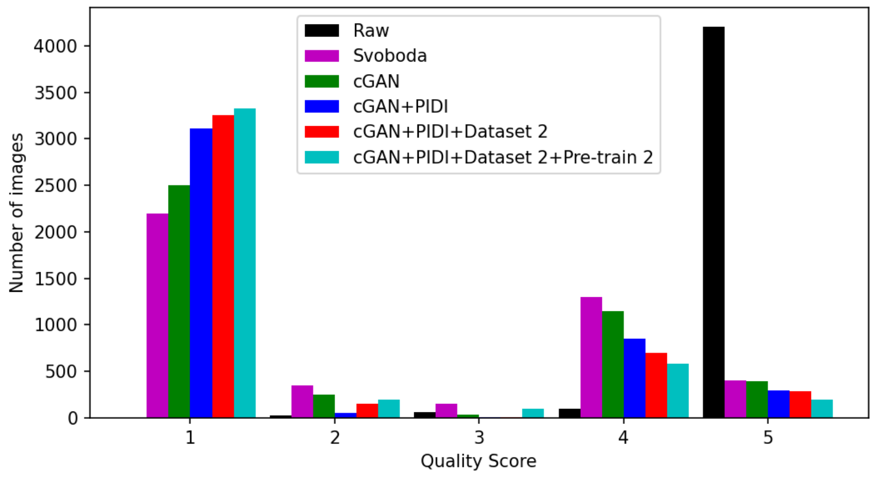


Figure 9.13: Quality assessment of the reconstructed samples using NFIQ.

9.3.4 Quality of the reconstructed fingerprints

Using the NFIQ utility from NBIS, the quality of reconstructed samples is measured to directly assess the performance of the reconstruction model. NFIQ assigns each fingerprint a numerical score from 1 (high quality) to 5 (low quality). Quality scores are computed for the reconstructed samples by our method and compared to score of both the raw latent fingerprints and those enhanced by the generative model developed by Svoboda et al. [16]. Figure 9.13 shows the quality scores of the reconstructed samples, and Fig. 9.7 shows three examples of the reconstructed fingerprints with different amount of distortion in the input sample. We again observe that both pre-training and the mixing augmentation improves the quality of the reconstructed ridge information.

9.4 Conclusion

Recognizing latent fingerprint samples is a challenging problem for identification systems since a latent fingerprint image can be ‘noisy’ with a large portions of the fingerprint missing, leading to a lower amount of ridge information compared to normal fingerprints. Following the successful outcomes of exploiting deep generative models for the traditional image processing problems, such as denoising, inpainting, and image to image translations, we propose

Table 9.4: Latent-to-Sensor matching using NBIS on the MOLF database.

		Accuracy (%)	
Sensor	Enhancement	Rank-25	Rank-50
Lumidigm	Raw	3.52	6.06
	Svoboda et al. [16]	16.71	23.03
	cGAN	28.82	36.07
	cGAN+PIDI	**40.52**	**64.80**
Secugen	Raw	5.48	9.19
	Svoboda et al. [16]	12.33	20.47
	cGAN	23.68	32.16
	cGAN+PIDI	**37.67**	**60.58**
CrossMatch	Raw	6.01	10.64
	Svoboda et al. [16]	14.39	22.73
	cGAN	28.37	35.23
	cGAN+PIDI	**37.61**	**62.55**

Table 9.5: Latent-to-Sensor matching using VeriFinger on the MOLF database.

		Accuracy (%)	
Sensor	Enhancement	Rank-25	Rank-50
Lumidigm	Raw	3.13	6.80
	Svoboda et al. [16]	19.51	26.24
	cGAN	30.47	39.38
	cGAN+PIDI	**42.04**	**70.89**
Secugen	Raw	2.33	6.37
	Svoboda et al. [16]	15.23	21.81
	cGAN	26.44	34.32
	cGAN+PIDI	**37.14**	**66.11**
CrossMatch	Raw	3.17	6.51
	Svoboda et al. [16]	18.34	24.78
	cGAN	28.30	37.31
	cGAN+PIDI	**41.27**	**68.61**

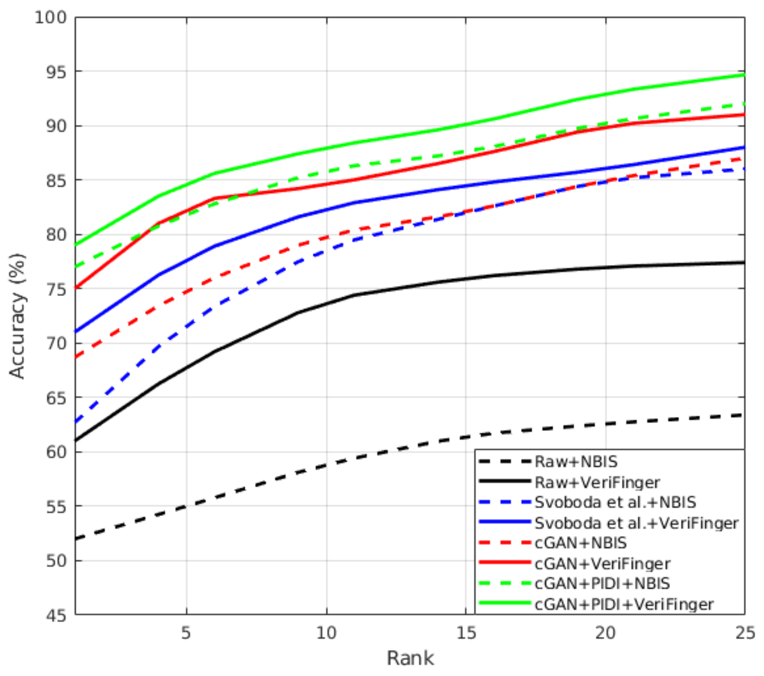


Figure 9.14: CMC curves for the experiment of latent-to-latent matching using NBIS and VeriFinger.

Table 9.6: Rank-1, rank-10 and rank-25 results for the experiment of latent-to-latent matching on the IIIT-Delhi latent database.

	Accuracy (%)		
Enhancement	Rank-1	Rank-10	Rank-25
Raw +NBIS	52.31	58.90	63.42
[16] +NBIS	62.69	78.85	86.12
cGAN +NBIS	68.69	79.85	87.23
cGAN+PIDI+NBIS	77.16	86.04	92.10
Raw +VeriFinger	61.02	74.00	77.44
[16] +VeriFinger	71.04	82.56	88.28
cGAN +VeriFinger	74.92	86.51	91.11
cGAN+PIDI+VeriFinger	**79.23**	**88.02**	**94.67**

a deep latent fingerprint reconstruction model based on conditional generative adversarial networks. We applied two modifications to the cGAN formulation and network architecture to adapt it for the task of latent fingerprint reconstruction. Generated ridge maps using GAN models often contain random ridge patterns for severely distorted areas of the input fingerprint. Main generator of our model is forced to generate three extra fingerprint maps. One of these maps is the ridge segmentation map which shows the reliability of the corresponding ridge map.

Opposed to the previous works in the literature, the proposed network directly translates the input latent fingerprints to the clean binary ridge maps by predicting the missing ridge information. Incorporating a discriminator network which measures both quality and PIDI of the reconstructed samples simultaneously, along with the generation process in the training phase, increases the quality of the generated samples directly without a need to define multiple complex loss functions for minimizing the similarity between the generated ridge patterns and the ground truth maps.

The proposed method successfully reconstructed latent fingerprints from the IIIT-Delhi latent and IIIT-Delhi MOLF databases in different experimental setups of latent-to-sensor and latent-to-latent matching. We achieved rank-50 accuracy of 70.89% for the latent-to-sensor matching on the IIIT-Delhi MOLF database. For the latent-to-latent matching we achieved rank-10 accuracy of 88.02%. Although the best results in both experiments were obtained when VeriFinger was used as the matcher, NBIS matching algorithm also resulted in high matching accuracy.

During our evaluations, we observed that the size and diversity of the training set has a major impact on the performance of reconstruction. However, due to the scarcity of the annotated latent fingerprints, it is not possible to gather a diverse dataset with abundant data. Hence, we considered two approaches to address this shortcoming. In the first approach, we developed a mixing augmentation technique to combine ridge information of several fingerprints to generate unseen training data and constructed a training set with more than a million synthetic latent fingerprints. In the second approach, we used unsupervised pre-training techniques of denoising autoencoder and contrastive learning to improve the knowledge of the model for the task of latent fingerprint reconstruction. Harnessing these two approaches, our model achieved 1.5% and 2.6% improvement compared to our original model using pre-training and the larger training set, respectively.

In addition, measuring the quality of reconstructed fingerprints using NFIQ shows that the generated fingerprints are significantly enhanced compared to the raw latent samples. For future work it is desired to directly extract minutiae from the latent input fingerprints. On the other hand, increasing the size of the synthetic latent database by introducing more complex distortions is another future direction for this work.

Chapter 10

Conclusion and Future Work

10.1 Conclusion

In this dissertation, we analyzed adversarial machine learning and its applications in computer vision and biometrics. In the first part of the dissertation, we investigated the susceptibility of DNNs to adversarial manipulation of the geometry and frequency-domain characteristics of the input examples. Then we proposed two methods for improving the robustness of DNNs against adversarial manipulations. In the second part, we developed methods for applications in computer vision and biometrics. Our conclusions on the analyses in each part of the dissertation are provided in the following.

Geometric Adversarial Faces. We empirically demonstrated that the geometry of the face is a major determinant for deep face recognition. According to this observation, we developed two adversarial attacks, termed FLM and GFLM, that manipulate the geometry of the face by tweaking landmark locations. FLM manipulates each landmark location independently and ignores the semantic structure of the face. Therefore, we observed distortions in the generated adversarial faces that increase the chance of detection by the human observer or a defense method. To address this, GFLM manipulates the landmark locations in a group fashion according to the semantic parts in the face. Our evaluations demonstrated that deep face recognition models are critically susceptible to geometric manipulations of the face. Furthermore, geometrically manipulated faces are closer to the manifold of natural faces compared to the pixel-intensity based adversarial attacks. Consequently, this attack is much more robust against conventional defense methods, especially the ones that built upon denoising or manifold learning.

Smooth Adversarial Perturbations. A major limitation of the conventional adversarial perturbations is their high-frequency pattern. This reduces the effectiveness of the perturbations in several ways. First, defenses can notably reduce the performance of the attack by performing denoising on the input examples before feeding them to the prediction model. Second, these perturbations are less realizable for real-world applications since printing devices often have a low-pass response. Third, they are less transferable to other models due to their sensitivity to simple transformations such as translation and rotation.

We developed an adversarial attack for crafting smooth adversarial perturbations. The proposed method is independent of the type of the smoothing kernel and the intensity of the smoothing. Through extensive evaluations, we observed that it is possible to craft smooth adversarial perturbations which alleviate the shortcomings of the common high-frequency perturbations. Furthermore, our analyses suggest that crafting smoother adversarial perturbations is more difficult than the conventional perturbations because DNNs intrinsically rely more on high-frequency patterns in the input examples.

Adversarial Robustness of Ensemble Models. Ensemble models have demonstrated improved performance compared to single models. We investigated the possibility of using ensemble predictions to improve the robustness against adversarial examples. Inspired by geometric theoretical analyses, we introduced a practical scenario of first-order defensive interactions between members of an ensemble. We both theoretically and empirically demonstrated that imposing these interactions significantly improves the robustness of ensembles. We proposed the joint gradient phase and magnitude regularization (GPMR) as an empirical tool to regularize the interaction between members and equalize their role in the ensemble decision. Furthermore, we concluded that the superior performance of GPMR is due to its capability to increase the effective number of members contributing to the robustness. We demonstrated that, unlike previous heuristic methods that diversify the predictions to improve the robustness, the gradients of the predictions must be diversified to achieve theoretically grounded robustness.

Revisiting Outer Optimization in Adversarial Training. Adversarial training methods generally adopt conventional momentum stochastic gradient descent (MSGD) for outer optimization. We demonstrated that the statistical characteristics of the gradients in adversarial training are fundamentally different than those in natural training. Hence, the common MSGD cannot provide the expected performance in adversarial training. To address this issue, we proposed a new optimization method which is more suitable for outer optimization in adversarial training. In particular, our proposed optimizer is less sensitive to the chaotic behavior of gradients. Through extensive evaluations, we validated the effectiveness of the proposed method for several benchmark approaches of adversarial training. Furthermore, our results suggest that regularizing the gradients in adversarial training alleviates its major shortcomings including robust overfitting and high sensitivity to hyperparameter setting.

Boosting Deep Face Recognition via Disentangling Appearance and Geometry. We proposed a novel approach for disentangling deep representations for the two major characteristics of the faces, appearance and geometry. The core idea of the was to construct geometrically identical faces by incorporating spatial transformations and exploiting their relative similarities to learn disentangled embedding representations. We demonstrated that the disentanglement provides two benefits for the training procedure of deep face recognition. First, it improves the generalization and training accuracy by geometrically augmenting the training set. Second, it enhances the learned knowledge of the early and intermediate layers of the deep model by enforcing them to satisfy the relative properties of appearance and geometry representations in the corresponding embedding spaces. We demonstrated that the knowledge learned through the disentangling approach can also be used to improve the performance of other face-related tasks, such as attribute prediction.

Supervised Mixing Data Augmentation. Mixing augmentation is a powerful approach to augment the training set and improve the generalization of the model at test time. However, the current mixing methods combine images in a blind way. This reduces the effectiveness of the mixing since salient features in input images can be deteriorated by averaging or overlapping with trivial features in other images. To address this issue, we proposed a supervised mixing augmentation method that combines input images based on their salient regions. The objective function of the proposed methods forces the mixed images to reside close to the manifold of natural images and also contain salient features of input images.

Through extensive evaluations on two tasks of object recognition and knowledge distillation, we demonstrate that the mixed images using the proposed approach notably improve the generalization of the model.

Fingerprint Distortion Rectification. We proposed a novel approach to estimate distortion parameters from raw fingerprint images without computing the ridge frequency and orientation maps. A deep convolutional neural network is utilized to estimate distortion parameters of input samples. We successfully rectified distorted samples from the Tsinghua DF database and FVC2004 DB1 using the estimated distortion template. A comprehensive database of distorted samples was generated in order to train our deep neural network. The experimental results on several databases showed that the proposed model can estimate the non-linear distortions of samples more accurately. Comparing to the previous works, our method reduced the rectification time significantly by embedding the training samples in the network parameters.

Latent Fingerprint Enhancement. Following the successful outcomes of utilizing deep generative models for the traditional image processing problems, such as denoising, inpainting, and image to image translations, we propose a deep latent fingerprint reconstruction model based on conditional generative adversarial networks. We applied two modifications to the cGAN formulation and network architecture to adapt it for the task of latent fingerprint reconstruction. In the first modification, we forced the main generator to generate three extra fingerprint maps to preserve the ridge structure of the input latent fingerprints. In the second modification, we developed a novel approach for preserving the ID information of latent fingerprints.

The proposed method successfully reconstructed latent fingerprints from the IIIT-Delhi latent and IIIT-Delhi MOLF databases in different experimental setups of latent-to-sensor and latent-to-latent matching. We achieved rank-50 accuracy of 70.89% for the latent-to-sensor matching on the IIIT-Delhi MOLF database. For the latent-to-latent matching we achieved rank-10 accuracy of 88.02%.

During our evaluations, we observed that the size and diversity of the training set has a major impact on the performance of reconstruction. However, due to the scarcity of the annotated latent fingerprints, it is not possible to gather a diverse dataset with abundant latent fingerprints. Hence, we considered two approaches to address this shortcoming. In the first approach, we developed a mixing augmentation technique to combine ridge information of several fingerprints to generate unseen training data and constructed a training set with more than a million synthetic latent fingerprints. In the second approach, we used unsupervised pre-training techniques of denoising autoencoder and contrastive learning to improve the knowledge of the model for the task of latent fingerprint reconstruction. Harnessing these two approaches, our model achieved 1.5% and 2.6% improvement compared to our original model using pre-training and the larger training set, respectively.

10.2 Future Work

In this section, we discuss different aspects of the current methods and applications developed in this dissertation that can be improved in the future or incorporated to other applications to enhance their performance.

Geometric Adversarial Faces. As demonstrated in Chapter 6, manipulating the geometry of the faces is a powerful approach for augmentation. Adversarial manipulation can provide the most challenging geometric augmentation since it directly seeks to maximize the identification loss. Hence, one major direction for the future research in this area is to adopt adversarial training on geometrically manipulated faces. This can be formalized by designing an adversarial game between the deep face recognition model and the adversary who aims to alter the face geometry to change the identity of the face. Geometric adversarial faces can also be used to augment the detests in other applications of face recognition such as face alignment and landmark detection, face morphing, face generation tasks using GANs, etc.

Smooth Adversarial Perturbations. In this work, we analyzed smooth adversarial perturbations for natural images. Another possible work in this direction can be to adopt the smooth perturbation for other tasks with different types of input signals such as voice recognition or depth estimation. Especially in voice recognition crafting smooth perturbations can be desirable for adversaries since the perturbation signal can be transmitted from an external source and there is no need to alter the main source of the signal. Furthermore, since natural phenomena such as lighting, shadow, and occlusions often have smooth frequency-domain characteristics, analyzing the robustness of machine learning based predictions to such effects can be beneficial for applications in the wild such as autonomous vehicles.

Adversarial Robustness of Ensemble Models. The current research on the robustness of ensemble models lacks an adversarial attack tailored specifically for the ensemble models. Hence, one future work in this direction can be to develop an adversarial attack for ensemble models. This attack can be generally built upon the well-known attack for single models. However, considering the ensemble predictions and local susceptibility of the members of the ensemble would be an interesting approach to focus on. Furthermore, adopting adversarial training as an approach for diversifying the gradients in the ensemble is another possible task for future investigation.

Outer Optimization in Adversarial Training. In this work, we proposed an approach for reducing the sensitivity of the optimizer to the chaotic behavior of gradients in adversarial training. Analyzing other approaches to control and regularized the gradients would be highly interesting. For instance, given supervision on the characteristics of the gradients, the inner optimizer can dynamically craft perturbations that cause limited overshoot in the statistics of the gradients. Furthermore, improving the computational efficiency of the proposed method is an important challenge that calls for fundamental research in this direction.

Disentangling Appearance and Geometry. In the current work, we constructed geometrically identical faces to disentangle the representations of the face. This disentanglement can be further improved by incorporating adversarial manipulations of the face geometry. Hence, while the model learns to disentangle the representations, it can simultaneously learn to discriminate between different manipulations of the face according to their impact on the recognition performance. Furthermore, another problem for future work can be to adopt the current approach to the general case of object recognition. To this aim, a discriminative model can provide the supervision to pair similar objects during the training and then transform them geometrically to construct the geometrically identical pairs for the disentanglement.

Mixing Augmentation. The current mixing augmentation method is evaluated in two tasks of object recognition and knowledge distillation. As future work, one can adopt the method to other computer vision and learning tasks such as object detection, semantic segmentation, key-point detection, un/semi-supervised learning, etc. Considerations according to the specifications of the underlying task can perhaps help improve the performance of the mixing augmentation. Furthermore, adopting the current approach for 3D tasks would be highly desirable since the size of 3D datasets is often limited compared to the huge size of the associated learning models.

Distortion Rectification. Since the estimation time of the proposed method is independent of the training size, it is possible to increase the number of principal components which are used to generate the synthetic distorted database for the future works. In this way, the proposed model can rectify distortions more accurately. In addition, by increasing the principal components of the distortion, the diversity of the synthetic dataset will be increased which eventually can lead to an improved performance.

Latent Fingerprint Enhancement. In our evaluations, we observed that the diversity of the training set can significantly contribute to the final reconstruction performance. However, due to the scarcity of annotated latent fingerprints, we generated synthetic latent fingerprints for the training of the model. The synthetic data is constructed by deliberately distorting clean fingerprints. A major direction for the future work can be to generate latent fingerprints using another generative adversarial network. In this way, the reconstruction model can be trained using latent fingerprints which may not exist in the real-world, but provide invaluable information regarding the distribution of ridge patterns in the real-world fingerprints. In addition, we observed that it is not possible to rectify the geometric distortion of latent fingerprints since the main generator has a UNet architecture. developing non-UNet architecture to address the shortcoming of UNet in correcting geometric distortion can be another prominent direction of research for the future work.

10.3 List of Publications

- SuperMix: Supervising the Mixing Data Augmentation, Dabouei, Soleymani, Taherkhani, and Nasrabadi, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.*

- Self-Supervised Wasserstein Pseudo-Labeling for Semi-Supervised Image Classification, Taherkhani, Dabouei, Soleymani, Dawson, Nasrabadi, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2021.*

- Quality-Aware Multimodal Biometric Recognition, Soleymani, Dabouei, Taherkhani, Iranmanesh, Dawson, Nasrabadi, *IEEE Transactions on Biometrics, Behavior, and Identity Science (TBIOM), 2021.*

- FDeblur-GAN: Fingerprint Deblurring Using Generative Adversarial Network, Joshi, Dabouei, Dawson, Nasrabadi, *IEEE International Joint Conference on Biometrics (IJCB), 2021.*

- Differential Morphed Face Detection Using Deep Siamese Networks, Soleymani, Chaudhary, Dabouei, Dawson, Nasrabadi, *International Conference on Pattern Recognition (ICPR), 2021.*

- Mutual information maximization on disentangled representations for differential morph detection, Soleymani, Dabouei, Taherkhani, Dawson, Nasrabadi, *In the IEEE Winter Conference on Applications of Computer Vision (WACV), 2021.*

- SmoothFool: An Efficient Framework for Computing Smooth Adversarial Perturbations, Dabouei, Soleymani, Taherkhani, Dawson, and Nasrabadi, *In the IEEE Winter Conference on Applications of Computer Vision (WACV), 2020.*

- Boosting Deep Face Recognition via Disentangling Appearance and Geometry, Dabouei, Taherkhani, Soleymani, Dawson, and Nasrabadi, *In the IEEE Winter Conference on Applications of Computer Vision (WACV), 2020.*

- Super-resolution Guided Pore Detection for Fingerprint Recognition, Ferdous, Dabouei, Dawson, Nasrabadi, *IEEE International Joint Conference on Biometrics (IJCB), 2020.*

- Efficient oct image segmentation using neural architecture search, Heidari Gheshlaghi, Dehzangi, Dabouei, Amireskandari, Rezai, Nasrabadi, *IEEE International Conference on Image Processing (ICIP), 2020.*

- Attribute adaptive margin softmax loss using privileged information, Iranmanesh, Dabouei, Nasrabadi, *The British Machine Vision Conference (BMVC), 2020.*
- Transporting labels via hierarchical optimal transport for semi-supervised learning, Taherkhani, Dabouei, Soleymani, Dawson, Nasrabadi, *European Conference on Computer Vision (ECCV), 2020.*
- Robust Facial Landmark Detection via Aggregation on Geometrically Manipulated Faces, Iranmanesh, Dabouei, Soleymani, Kazemi, Nasrabadi, *In the IEEE Winter Conference on Applications of Computer Vision (WACV), 2020.*
- Exploiting Joint Robustness to Adversarial Perturbations, Dabouei, Soleymani, Taherkhani, Dawson, and Nasrabadi, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.*
- Quality Guided Sketch-to-Photo Image Synthesis, Osahor, Kazemi, Dabouei, Nasrabadi, *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW), 2020.*
- Fast Geometrically-perturbed Adversarial Faces, Dabouei, Soleymani, Dawson, and Nasrabadi, *In the IEEE Winter Conference on Applications of Computer Vision (WACV), 2019.*
- Defending Against Adversarial Iris Examples Using Wavelet Decomposition, Soleymani, Dabouei, Dawson, Nasrabadi, *IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS), 2019.*
- Adversarial Examples to Fool Iris Recognition Systems, Soleymani, Dabouei, Dawson, Nasrabadi, *International Conference on Biometrics (ICB), 2019.*
- Deep Contactless Fingerprint Unwarping, Dabouei, Soleymani, Dawson, Nasrabadi, *International Conference on Biometrics (ICB), 2019.*
- A Weakly Supervised Fine Label Classifier Enhanced by Coarse Supervision, Taherkhani, Kazemi, Dabouei, Dawson, Nasrabadi, *Proceedings of the IEEE/CVF International*

Conference on Computer Vision (ICCV), 2019.

- Prosodic-Enhanced Siamese Convolutional Neural Networks for Cross-Device Text-Independent Speaker Verification, Soleymani, Dabouei, Iranmanesh, Kazemi, Dawson, Nasrabadi, *IEEE 9th international conference on biometrics theory, applications and systems (BTAS), 2018.*
- ID Preserving Generative Adversarial Network for Partial Latent Fingerprint Reconstruction, Dabouei, Soleymani, Kazemi, Iranmanesh, Dawson, Nasrabadi, *IEEE 9th international conference on biometrics theory, applications and systems (BTAS), 2018.*
- Multi-Level Feature Abstraction From Convolutional Neural Networks for Multimodal Biometric Identification, Soleymani, Dabouei, Kazemi, Dawson, Nasrabadi, *International Conference on Pattern Recognition (ICPR), 2018.*
- Facial Attributes Guided Deep Sketch-to-Photo Synthesis, Kazemi, Iranmanesh, Dabouei, Soleymani, Nasrabadi, *IEEE Winter Applications of Computer Vision Workshops (WACVW), 2018.*
- Deep Cross Polarimetric Thermal-to-Visible Face Recognition, Kazemi, Iranmanesh, Dabouei, Soleymani, Nasrabadi, *International Conference on Biometrics (ICB), 2018.*
- Fingerprint Distortion Rectification Using Deep Convolutional Neural Networks, Dabouei, Kazemi, Iranmanesh, Dawson, Nasrabadi, *International Conference on Biometrics (ICB), 2018.*

References

[1] C. Xiao, J.-Y. Zhu, B. Li, W. He, M. Liu, and D. Song, "Spatially transformed adversarial examples," *arXiv preprint arXiv:1801.02612*, 2018. x, 7, 8, 10, 13, 14, 21, 22, 23, 43

[2] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples (2014)," *arXiv preprint arXiv:1412.6572*, 2014. x, xiii, 6, 7, 9, 10, 11, 14, 22, 23

[3] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "DeepFool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582. x, xiv, 6, 25, 26, 27, 31, 32, 37, 39, 47, 49, 52, 56, 64, 71, 87, 115

[4] 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 *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.* ACM, 2016, pp. 1528–1540. x, 2, 11, 14, 26, 28, 37, 39

[5] 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 visual classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1625–1634. x, 39

[6] H. Hosseini and R. Poovendran, "Semantic adversarial examples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1614–1619. x, 28, 37, 39, 42

[7] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database forstudying face recognition in unconstrained environments," in *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*, 2008. xi, xv, xvi, 17, 98, 100, 102, 104

[8] X. Si, J. Feng, J. Zhou, and Y. Luo, "Detection and rectification of distorted fingerprints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 555–568, March 2015. xii, xvi, 134, 136, 137, 139, 140, 141, 142

[9] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456. xii, 151, 154, 155

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations (ICLR)*, 2015. xiii, 2, 25, 27, 28, 43, 47, 48, 49, 51, 52, 56, 65

[11] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016. xiv, 6, 10, 25, 26, 27, 33, 49, 56, 65, 70

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778. xiv, 1, 26, 36, 90, 98

[13] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *International Conference on Machine Learning*. PMLR, 2020, pp. 2206–2216. xv, 71, 80, 85

[14] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929. xvi, 116

[15] A. Sankaran, M. Vatsa, R. Singh, and A. Majumdar, "Group sparse autoencoder," *Image and Vision Computing*, vol. 60, pp. 64–74, 2017. xvi, 145, 146

[16] J. Svoboda, F. Monti, and M. M. Bronstein, "Generative convolutional networks for latent fingerprint reconstruction," *arXiv preprint arXiv:1705.01707*, 2017. xvi, 139, 145, 146, 164, 165, 166

[17] A. Sankaran, M. Vatsa, and R. Singh, "Latent fingerprint matching: A survey." *IEEE Access*, vol. 2, no. 982-1004, p. 1, 2014. xvii, 158

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105. 1, 6, 25, 47

[19] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014. 1

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017. 1

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018. 1

[22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, 2017, pp. 2261–2269. 1

[23] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention.* Springer, 2015, pp. 234–241. 1, 151

[24] D. P. Kingma and J. Ba, "ADAM: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 1

[25] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016. 1

[26] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of Machine Learning Research*, vol. 12, no. 7, 2011. 1

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680. 1, 2, 147

[28] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint*, 2017. 1, 2

[29] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition.* Springer, 2015, pp. 84–92. 1

[30] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "SphereFace: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 212–220. 1, 90, 91, 92, 97, 101, 102, 103, 104

[31] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699. 1

[32] 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. 2, 6, 9, 11, 25, 47, 48, 49, 56, 70

[33] T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: Identifying vulnerabilities in the machine learning model supply chain," *arXiv preprint arXiv:1708.06733*, 2017. 2

[34] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," *arXiv preprint arXiv:1801.01944*, 2018. 2, 25

[35] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial examples for malware detection," in *European Symposium on Research in Computer Security.* Springer, 2017, pp. 62–79. 2, 25

[36] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial attacks on neural network policies," *arXiv preprint arXiv:1702.02284*, 2017. 2, 25

[37] C. Sitawarin, A. N. Bhagoji, A. Mosenia, M. Chiang, and P. Mittal, "Darts: Deceiving autonomous cars with toxic signs," *arXiv preprint arXiv:1802.06430*, 2018. 2

[38] A. Dabouei, S. Soleymani, J. Dawson, and N. Nasrabadi, "Fast geometrically-perturbed adversarial faces," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV).* IEEE, 2019, pp. 1979–1988. 2, 43, 47, 92

[39] A. Shafahi, W. R. Huang, C. Studer, S. Feizi, and T. Goldstein, "Are adversarial examples inevitable?" in *International Conference on Learning Representations (ICLR)*, 2019. 2, 25

[40] M. Hein and M. Andriushchenko, "Formal guarantees on the robustness of a classifier against adversarial manipulation," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 2266–2276. 2, 25, 48, 49, 52, 56

[41] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014. 2

[42] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014. 3, 70

[43] 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. 3, 11, 22, 23, 70, 71, 83

[44] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "MixUp: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017. 5, 110, 111, 112, 113

[45] Y. Tokozume, Y. Ushiku, and T. Harada, "Between-class learning for image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5486–5494. 5, 110, 111, 112

[46] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "CutMix: Regularization strategy to train strong classifiers with localizable features," *arXiv preprint arXiv:1905.04899*, 2019. 5, 110, 111, 113

[47] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *arXiv preprint*, 2017. 6

[48] S. Mohamadi, N. M. Nasrabadi, G. Doretto, and D. A. Adjeroh, "Human age estimation from gene expression data using artificial neural networks," in *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* IEEE, 2021, pp. 3492–3497. 6, 146

[49] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *arXiv preprint arXiv:1412.5068*, 2014. 6

[50] F. Liao, M. Liang, Y. Dong, T. Pang, X. Hu, and J. Zhu, "Defense against adversarial attacks using high-level representation guided denoiser," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7, 10, 26, 28, 43

[51] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision.* Springer, 2016, pp. 694–711. 7, 150

[52] W. Wang, A. Wang, A. Tamar, X. Chen, and P. Abbeel, "Safer classification by synthesis," *arXiv preprint arXiv:1711.08534*, 2017. 8

[53] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, 1989. 9, 11

[54] B. Liang, H. Li, M. Su, X. Li, W. Shi, and X. Wang, "Detecting adversarial examples in deep networks with adaptive noise reduction," *arXiv preprint arXiv:1705.08378*, 2017. 10, 11

[55] A. Rozsa, E. M. Rudd, and T. E. Boult, "Adversarial diversity and hard positive generation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 25–32. 10

[56] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li, "Boosting adversarial attacks with momentum," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9185–9193. 10, 47, 65, 71, 80

[57] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P).* IEEE, 2016, pp. 372–387. 10, 25, 65, 80

[58] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 IEEE Symposium on Security and Privacy (SP).* IEEE, 2017, pp. 39–57. 10, 25, 47, 49, 65, 71, 80

[59] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025. 10, 18

[60] A. Harvey, "CV dazzle: Camouflage from face detection," *Master's Thesis, New York University*, 2017. 11

[61] A. Goel, A. Singh, A. Agarwal, M. Vatsa, and R. Singh, "Smartbox: Benchmarking adversarial detection and mitigation algorithms for face recognition," in *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS).* IEEE, 2018, pp. 1–7. 11

[62] G. Goswami, N. Ratha, A. Agarwal, R. Singh, and M. Vatsa, "Unravelling robustness of deep learning based face recognition against adversarial attacks," *arXiv preprint arXiv:1803.00401*, 2018. 11

[63] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," *arXiv preprint arXiv:1705.07204*, 2017. 11, 22, 23

[64] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau, "Keeping the bad guys out: Protecting and vaccinating deep learning with JPEG compression," *arXiv preprint arXiv:1705.02900*, 2017. 11

[65] Q. Wang, W. Guo, K. Zhang, I. Ororbia, G. Alexander, X. Xing, X. Liu, and C. L. Giles, "Learning adversary-resistant deep neural networks," *arXiv preprint arXiv:1612.01401*, 2016. 11

[66] M. O. Irfanoglu, B. Gokberk, and L. Akarun, "3d shape-based face recognition using automatically registered facial surfaces," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 4. IEEE, 2004, pp. 183–186. 13

[67] O. M. Parkhi, A. Vedaldi, A. Zisserman *et al.*, "Deep face recognition," in *Proceedings of the British Machine Vision Conference (BMVC)*, vol. 1, no. 3, 2015, p. 6. 14, 89, 90, 92, 102

[68] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823. 17, 90, 91, 92, 102, 103

[69] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "VGGFace2: A dataset for recognising faces across pose and age," in *Automatic Face & Gesture Recognition (FG 2018), 2018 13th IEEE International Conference on.* IEEE, 2018, pp. 67–74. 17, 101

[70] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014. 17, 18, 22, 89, 99, 100

[71] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1755–1758, 2009. 18, 99

[72] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 567–585, 1989. 18, 94

[73] Y. Zhang, K. Lee, and H. Lee, "Augmenting supervised neural networks with unsupervised objectives for large-scale image classification," in *International Conference on Machine Learning (ICML)*, 2016, pp. 612–621. 25

[74] F. Taherkhani, H. Kazemi, A. Dabouei, J. Dawson, and N. M. Nasrabadi, "A weakly supervised fine label classifier enhanced by coarse supervision," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6459–6468. 25, 90

[75] F. Taherkhani, H. Kazemi, and N. M. Nasrabadi, "Matrix completion for graph-based deep semi-supervised learning," in *Thirty-Third AAAI Conference on Artificial Intelligence*, 2019. 25, 90

[76] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," in *International Conference on Learning Representations (ICLR)*, 2018. 25, 26, 28, 43

[77] V. Fischer, M. C. Kumar, J. H. Metzen, and T. Brox, "Adversarial examples for semantic image segmentation," *arXiv preprint arXiv:1703.01101*, 2017. 25

[78] J. Hendrik Metzen, M. Chaithanya Kumar, T. Brox, and V. Fischer, "Universal adversarial perturbations against semantic image segmentation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2755–2764. 25

[79] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, "Adversarial examples for semantic segmentation and object detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 25

[80] D. Song, K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, F. Tramer, A. Prakash, and T. Kohno, "Physical adversarial examples for object detectors," in *12th {USENIX} Workshop on Offensive Technologies ({WOOT} 18)*, 2018. 25

[81] J. Su, D. V. Vargas, and S. Kouichi, "One pixel attack for fooling deep neural networks," *arXiv preprint arXiv:1710.08864*, 2017. 25

[82] A. Modas, S.-M. Moosavi-Dezfooli, and P. Frossard, "SparseFool: a few pixels make a big difference," *arXiv preprint arXiv:1811.02248*, 2018. 25, 32, 33

[83] S. T. Jan, J. Messou, Y.-C. Lin, J.-B. Huang, and G. Wang, "Connecting the digital and physical world: Improving the robustness of adversarial attacks," in *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI'19)*, 2019. 26

[84] S.-M. Moosavi-Dezfooli, A. Shrivastava, and O. Tuzel, "Divide, denoise, and defend against adversarial attacks," *arXiv preprint arXiv:1802.06806*, 2018. 26

[85] A. Prakash, N. Moran, S. Garber, A. DiLillo, and J. Storer, "Deflecting adversarial attacks with pixel deflection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8571–8580. 26

[86] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations (ICLR)*, 2018. 26, 28, 43, 48, 49, 52, 56, 65

[87] P. Samangouei, M. Kabkab, and R. Chellappa, “Defense-GAN: Protecting classifiers against adversarial attacks using generative models,” in *International Conference on Learning Representations (ICLR)*, 2018. 26, 49

[88] Y. Dong, T. Pang, H. Su, and J. Zhu, “Evading defenses to transferable adversarial examples by translation-invariant attacks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4312–4321. 26, 28, 44

[89] A. Fawzi, S. M. Moosavi Dezfooli, and P. Frossard, “The robustness of deep networks-a geometric perspective,” *IEEE Signal Processing Magazine*, vol. 34, 2017. 27, 31, 48, 51

[90] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, and S. Soatto, “Empirical study of the topology and geometry of deep networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 27, 48, 51

[91] R. C. Fong and A. Vedaldi, “Interpretable explanations of black boxes by meaningful perturbation,” in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 3449–3457. 28, 115, 127

[92] D. Meng and H. Chen, “MagNet: a two-pronged defense against adversarial examples,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 135–147. 29, 43, 49

[93] S. Jetley, N. Lord, and P. Torr, “With friends like these, who needs adversaries?” in *Advances in Neural Information Processing Systems (NIPS)*, 2018, pp. 10 772–10 782. 31

[94] Y. LeCun, C. Cortes, and C. Burges, “MNIST handwritten digit database,” *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, vol. 2, 2010. 36

[95] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” Citeseer, Tech. Rep., 2009. 36, 79, 109, 111, 120

[96] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 2009, pp. 248–255. 36, 120

[97] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. 36

[98] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, “Return of the devil in the details: Delving deep into convolutional nets,” *arXiv preprint arXiv:1405.3531*, 2014. 36

[99] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1765–1773. 44, 47

[100] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 47

[101] K. R. Mopuri, U. Garg, and R. V. Babu, "Fast feature fool: A data independent approach to universal adversarial perturbations," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2017. 47

[102] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *International Conference on Machine Learning*, 2018, pp. 284–293. 47

[103] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," *arXiv preprint arXiv:1712.09665*, 2017. 47

[104] A. Dabouei, S. Soleymani, F. Taherkhani, J. Dawson, and N. Nasrabadi, "Smoothfool: An efficient framework for computing smooth adversarial perturbations," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 2665–2674. 47

[105] N. Ford, J. Gilmer, N. Carlini, and D. Cubuk, "Adversarial examples are a natural consequence of test error in noise," *arXiv preprint arXiv:1901.10513*, 2019. 47, 48, 49, 70

[106] J. M. Cohen, E. Rosenfeld, and J. Z. Kolter, "Certified adversarial robustness via randomized smoothing," *arXiv preprint arXiv:1902.02918*, 2019. 47, 48, 70

[107] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu, "Improving adversarial robustness via promoting ensemble diversity," *arXiv preprint arXiv:1901.08846*, 2019. 48, 49, 50, 52, 55, 58, 65, 67

[108] S. Kariyappa and M. K. Qureshi, "Improving adversarial robustness of ensembles with diversity training," *arXiv preprint arXiv:1901.09981*, 2019. 48, 49, 50, 52, 54, 55, 58

[109] M. Abbasi and C. Gagné, "Robustness to adversarial examples through an ensemble of specialists," *arXiv preprint arXiv:1702.06856*, 2017. 48, 49, 52, 55

[110] A. Bagnall, R. Bunescu, and G. Stewart, "Training ensembles to detect adversarial examples," *arXiv preprint arXiv:1712.04006*, 2017. 48, 49, 50, 52, 55

[111] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE Transactions on Neural Networks and Learning Systems*, 2019. 49

[112] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 582–597. 49

[113] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks," *arXiv preprint arXiv:1704.01155*, 2017. 49

[114] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy, "A study of the effect of JPG compression on adversarial images," *arXiv preprint arXiv:1608.00853*, 2016. 49

[115] Z. Zheng and P. Hong, "Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 7913–7922. 49

[116] N. Carlini and D. Wagner, "Adversarial examples are not easily detected: Bypassing ten detection methods," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 2017, pp. 3–14. 49

[117] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier, "Parseval networks: Improving robustness to adversarial examples," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 854–863. 49, 52, 56

[118] Y. Tsuzuku, I. Sato, and M. Sugiyama, "Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 6541–6550. 49, 52, 56

[119] Z. Yan, Y. Guo, and C. Zhang, "Deep defense: Training DNNs with improved adversarial robustness," in *Advances in Neural Information Processing Systems*, 2018, pp. 419–428. 49

[120] O. L. Mangasarian, "Arbitrary-norm separating plane," *Operations Research Letters*, vol. 24, no. 1-2, pp. 15–23, 1999. 51

[121] X. V. Doan and S. Vavasis, "Finding approximately rank-one submatrices with the nuclear norm and \ell_1-norm," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2502–2540, 2013. 51

[122] O. L. Mangasarian, *Nonlinear Programming*. SIAM, 1994. 53

[123] A. S. Ross and F. Doshi-Velez, "Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 56

[124] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "EAD: elastic-net attacks to deep neural networks via adversarial examples," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 65

[125] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016. 65

[126] A. Kurakin, I. Goodfellow, S. Bengio, Y. Dong, F. Liao, M. Liang, T. Pang, J. Zhu, X. Hu, C. Xie *et al.*, "Adversarial attacks and defences competition," in *The NIPS'17 Competition: Building Intelligent Systems*. Springer, 2018, pp. 195–231. 67

[127] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” *arXiv preprint arXiv:1802.00420*, 2018. 67, 71

[128] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, “Deepdriving: Learning affordance for direct perception in autonomous driving,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730. 70

[129] X. Ma, Y. Niu, L. Gu, Y. Wang, Y. Zhao, J. Bailey, and F. Lu, “Understanding adversarial attacks on deep learning based medical image analysis systems,” *Pattern Recognition*, vol. 110, p. 107332, 2021. 70

[130] P. Nakkiran, “Adversarial robustness may be at odds with simplicity,” *arXiv preprint arXiv:1901.00532*, 2019. 71

[131] J. Zhang, X. Xu, B. Han, G. Niu, L. Cui, M. Sugiyama, and M. Kankanhalli, “Attacks which do not kill training make adversarial learning stronger,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 11 278–11 287. 71, 80, 83, 85

[132] C. Sitawarin, S. Chakraborty, and D. Wagner, “Improving adversarial robustness through progressive hardening,” *arXiv preprint arXiv:2003.09347*, 2020. 71, 80, 83, 85

[133] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, “Theoretically principled trade-off between robustness and accuracy,” in *International Conference on Machine Learning*. PMLR, 2019, pp. 7472–7482. 71, 79, 80, 82, 83, 85

[134] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, and Q. Gu, “Improving adversarial robustness requires revisiting misclassified examples,” in *International Conference on Learning Representations*, 2019. 71, 79, 83

[135] T. Pang, H. Zhang, D. He, Y. Dong, H. Su, W. Chen, J. Zhu, and T.-Y. Liu, “Adversarial training with rectified rejection,” *arXiv preprint arXiv:2105.14785*, 2021. 71

[136] A. Shafahi, M. Najibi, A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, “Adversarial training for free!” *arXiv preprint arXiv:1904.12843*, 2019. 71

[137] H. Zheng, Z. Zhang, J. Gu, H. Lee, and A. Prakash, “Efficient adversarial training with transferable adversarial examples,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1181–1190. 71

[138] E. Wong, L. Rice, and J. Z. Kolter, “Fast is better than free: Revisiting adversarial training,” *arXiv preprint arXiv:2001.03994*, 2020. 71

[139] L. Rice, E. Wong, and Z. Kolter, “Overfitting in adversarially robust deep learning,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 8093–8104. 71, 80

[140] T. Chen, Z. Zhang, S. Liu, S. Chang, and Z. Wang, "Robust overfitting may be mitigated by properly learned smoothening," in *International Conference on Learning Representations*, 2020. 71

[141] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization (2016)," *arXiv preprint arXiv:1611.03530*, 2017. 71

[142] B. Neyshabur, S. Bhojanapalli, D. Mcallester, and N. Srebro, "Exploring generalization in deep learning," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5947–5956, 2017. 71

[143] M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias–variance trade-off," *Proceedings of the National Academy of Sciences*, vol. 116, no. 32, pp. 15 849–15 854, 2019. 71

[144] S. Gowal, C. Qin, J. Uesato, T. Mann, and P. Kohli, "Uncovering the limits of adversarial training against norm-bounded adversarial examples," *arXiv preprint arXiv:2010.03593*, 2020. 71, 79, 87

[145] T. Pang, X. Yang, Y. Dong, H. Su, and J. Zhu, "Bag of tricks for adversarial training," *arXiv preprint arXiv:2010.00467*, 2020. 71, 79, 80, 87

[146] D. Wu, S.-T. Xia, and Y. Wang, "Adversarial weight perturbation helps robust generalization," *arXiv preprint arXiv:2004.05884*, 2020. 71, 79, 80, 83, 84, 85

[147] F. He, T. Liu, and D. Tao, "Control batch size and learning rate to generalize well: Theoretical and empirical evidence," *Advances in Neural Information Processing Systems*, vol. 32, pp. 1143–1152, 2019. 72, 74

[148] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," *arXiv preprint arXiv:1908.03265*, 2019. 72, 74

[149] S.-Y. Zhao, Y.-P. Xie, and W.-J. Li, "Stochastic normalized gradient descent with momentum for large batch training," *arXiv preprint arXiv:2007.13985*, 2020. 72, 75

[150] H. Yu, R. Jin, and S. Yang, "On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization," in *International Conference on Machine Learning.* PMLR, 2019, pp. 7184–7193. 74, 75

[151] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964. 75

[152] Y. Yan, T. Yang, Z. Li, Q. Lin, and Y. Yang, "A unified analysis of stochastic momentum methods for deep learning," *arXiv preprint arXiv:1808.10396*, 2018. 75

[153] Y. Le and X. Yang, "Tiny ImageNet visual recognition challenge," *CS 231N, Course Notes, Stanford University*, vol. 7, no. 7, p. 3, 2015. 79

[154] J. Cui, S. Liu, L. Wang, and J. Jia, "Learnable boundary guided adversarial training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 721–15 730. 79, 80, 83, 85

[155] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016. 79

[156] J. Uesato, B. O'donoghue, P. Kohli, and A. Oord, "Adversarial risk and the dangers of evaluating against weak attacks," in *International Conference on Machine Learning.* PMLR, 2018, pp. 5025–5034. 80

[157] L. Huang, C. Zhang, and H. Zhang, "Self-adaptive training: beyond empirical risk minimization," *Advances in Neural Information Processing Systems*, vol. 33, 2020. 80, 83, 85

[158] J. Chen, Y. Cheng, Z. Gan, Q. Gu, and J. Liu, "Efficient robust training via backward smoothing," *arXiv preprint arXiv:2010.01278*, 2020. 80, 83, 85

[159] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013. 80

[160] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International Conference on Machine Learning.* PMLR, 2013, pp. 1310–1318. 80

[161] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," in *International Conference on Learning Representations*, 2018. 82

[162] S.-M. Moosavi-Dezfooli, A. Fawzi, J. Uesato, and P. Frossard, "Robustness via curvature regularization, and vice versa," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9078–9086. 85

[163] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 12, pp. 2037–2041, 2006. 89, 92

[164] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "EigenFaces vs. FisherFaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 7, pp. 711–720, 1997. 89

[165] Y. Duan, J. Lu, J. Feng, and J. Zhou, "Context-aware local binary feature learning for face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 5, pp. 1139–1153, 2018. 89

[166] Z. Lei, M. Pietikäinen, and S. Z. Li, "Learning discriminant face descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 2, pp. 289–302, 2013. 89

[167] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, “MS-Celeb-1M: A dataset and benchmark for large-scale face recognition,” in *European Conference on Computer Vision.* Springer, 2016, pp. 87–102. 89, 101

[168] Y. Sun, Y. Chen, X. Wang, and X. Tang, “Deep learning face representation by joint identification-verification,” in *Advances in Neural Information Processing Systems*, 2014, pp. 1988–1996. 90, 91, 92

[169] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5265–5274. 90, 91, 92, 97, 101, 102, 103

[170] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, “A discriminative feature learning approach for deep face recognition,” in *European Conference on Computer Vision.* Springer, 2016, pp. 499–515. 91, 92, 96, 103

[171] C. Liu and H. Wechsler, “Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition,” *IEEE Transactions on Image Processing*, vol. 11, no. 4, pp. 467–476, 2002. 92

[172] Z. Cao, Q. Yin, X. Tang, and J. Sun, “Face recognition with learning-based descriptor,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* IEEE, 2010, pp. 2707–2714. 92

[173] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “DeepFace: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708. 92, 102

[174] F. Galton, “Personal identification and description,” *Journal of Anthropological Institute of Great Britain and Ireland*, pp. 177–191, 1889. 92

[175] S. M. Iranmanesh, A. Dabouei, S. Soleymani, H. Kazemi, and N. M. Nasrabadi, “Robust facial landmark detection via aggregation on geometrically manipulated faces,” *arXiv preprint arXiv:2001.03113*, 2020. 92

[176] A. K. Jain and U. Park, “Facial marks: Soft biometric for face recognition,” in *2009 16th IEEE International Conference on Image Processing (ICIP).* IEEE, 2009, pp. 37–40. 93

[177] U. Park and A. K. Jain, “Face matching and retrieval using soft biometrics,” *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 3, pp. 406–415, 2010. 93

[178] Z. Shu, M. Sahasrabudhe, R. Alp Guler, D. Samaras, N. Paragios, and I. Kokkinos, “Deforming autoencoders: Unsupervised disentangling of shape and appearance,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 650–665. 93

[179] X. Xing, R. Gao, T. Han, S.-C. Zhu, and Y. N. Wu, "Deformable generator network: Unsupervised disentanglement of appearance and geometry," *arXiv preprint arXiv:1806.06298*, 2018. 93

[180] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, "The MegaFace benchmark: 1 million faces for recognition at scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4873–4882. 100, 102

[181] J. Liu, Y. Deng, T. Bai, Z. Wei, and C. Huang, "Targeting ultimate accuracy: Face recognition via deep embedding," *arXiv preprint arXiv:1506.07310*, 2015. 102

[182] Z. Wang, K. He, Y. Fu, R. Feng, Y.-G. Jiang, and X. Xue, "Multi-task deep neural network for joint face recognition and facial attribute prediction," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval.* ACM, 2017, pp. 365–374. 103

[183] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 2011, pp. 529–534. 102

[184] H.-W. Ng and S. Winkler, "A data-driven approach to cleaning large face datasets," in *2014 IEEE International Conference on Image Processing (ICIP).* IEEE, 2014, pp. 343–347. 102

[185] N. Kumar, P. Belhumeur, and S. Nayar, "FaceTracer: A search engine for large collections of images with faces," in *European Conference on Computer Vision.* Springer, 2008, pp. 340–353. 105, 107

[186] N. Zhang, M. Paluri, M. Ranzato, T. Darrell, and L. Bourdev, "Panda: Pose aligned networks for deep attribute modeling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1637–1644. 105, 107

[187] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3730–3738. 105, 106, 107

[188] M. M. Kalayeh, B. Gong, and M. Shah, "Improving facial attribute prediction using semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6942–6950. 106

[189] F. Taherkhani, N. M. Nasrabadi, and J. Dawson, "A deep face identification network enhanced by facial attributes prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 553–560. 106

[190] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "DropOut: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014. 109

[191] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016. 109

[192] L. Kang, P. Ye, Y. Li, and D. Doermann, "Convolutional neural networks for no-reference image quality assessment," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1733–1740. 109

[193] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry, "Adversarially robust generalization requires more data," in *Advances in Neural Information Processing Systems*, 2018, pp. 5014–5026. 109

[194] J. Hestness, S. Narang, N. Ardalani, G. Diamos, H. Jun, H. Kianinejad, M. Patwary, M. Ali, Y. Yang, and Y. Zhou, "Deep learning scaling is predictable, empirically," *arXiv preprint arXiv:1712.00409*, 2017. 109

[195] D. Han, J. Kim, and J. Kim, "Deep pyramidal residual networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5927–5935. 109, 111

[196] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017. 109, 111

[197] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "AutoAugment: Learning augmentation strategies from data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 113–123. 109, 111, 118, 121

[198] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast AutoAugment," in *Advances in Neural Information Processing Systems*, 2019, pp. 6662–6672. 109, 111, 118, 121

[199] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "RandAugment: Practical data augmentation with no separate search," *arXiv preprint arXiv:1909.13719*, 2019. 109, 111, 118, 121

[200] J. Lemley, S. Bazrafkan, and P. Corcoran, "Smart augmentation learning an optimal data augmentation strategy," *IEEE Access*, vol. 5, pp. 5858–5869, 2017. 110, 111

[201] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *arXiv preprint arXiv:1712.04621*, 2017. 110

[202] H. Guo, Y. Mao, and R. Zhang, "MixUp as locally linear out-of-manifold regularization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 3714–3722. 110, 111

[203] H. Scudder, "Probability of error of some adaptive pattern-recognition machines," *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965. 110

[204] V. Vapnik and V. Vapnik, *Statistical Learning Theory*, 1998. 110

[205] C. Rosenberg, M. Hebert, and H. Schneiderman, "Semi-supervised self-training of object detection models." *WACV/MOTION*, vol. 2, 2005. 110

[206] L.-J. Li and L. Fei-Fei, “Optimol: automatic online picture collection via incremental model learning,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 147–168, 2010. 110

[207] X. Chen, A. Shrivastava, and A. Gupta, “Neil: Extracting visual knowledge from web data,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1409–1416. 110

[208] D. Yarowsky, “Unsupervised word sense disambiguation rivaling supervised methods,” in *33rd Annual Meeting of the Association for Computational Linguistics*, 1995, pp. 189–196. 110

[209] C. Bucilu, R. Caruana, and A. Niculescu-Mizil, “Model compression,” in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 2006, pp. 535–541. 110, 120

[210] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015. 110, 119, 120, 121, 122, 123, 128

[211] D. Ho, E. Liang, I. Stoica, P. Abbeel, and X. Chen, “Population based augmentation: Efficient learning of augmentation policy schedules,” *arXiv preprint arXiv:1905.05393*, 2019. 111, 121

[212] P. Dabkowski and Y. Gal, “Real time image saliency for black box classifiers,” in *Advances in Neural Information Processing Systems*, 2017, pp. 6967–6976. 115

[213] A. P. Ruszczyński and A. Ruszczynski, *Nonlinear Optimization.* Princeton university press, 2006, vol. 13. 115

[214] A. Dabouei, S. Soleymani, F. Taherkhani, J. Dawson, and N. M. Nasrabadi, “Smoothfool: An efficient framework for computing smooth adversarial perturbations,” *arXiv preprint arXiv:1910.03624*, 2019. 115

[215] Y. Tian, D. Krishnan, and P. Isola, “Contrastive representation distillation,” *arXiv preprint arXiv:1910.10699*, 2019. 119, 120, 121, 123

[216] P. Chrabaszcz, I. Loshchilov, and F. Hutter, “A downsampled variant of ImageNet as an alternative to the CIFAR datasets,” *arXiv preprint arXiv:1707.08819*, 2017. 120

[217] R. Cappelli, D. Maio, D. Maltoni, J. L. Wayman, and A. K. Jain, “Performance evaluation of fingerprint verification systems,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 1, pp. 3–18, 2006. 130

[218] R. Cappelli, M. Ferrara, A. Franco, and D. Maltoni, “Fingerprint verification competition 2006,” *Biometric Technology Today*, vol. 15, no. 7, pp. 7–9, 2007. 131

[219] L. M. Wein and M. Baveja, “Using fingerprint image quality to improve the identification performance of the us visitor and immigrant status indicator technology program,” *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 21, pp. 7772–7775, 2005. 131

[220] J. Feng, A. K. Jain, and A. Ross, "Detecting altered fingerprints," in *Pattern Recognition (ICPR), 2010 20th International Conference on.* IEEE, 2010, pp. 1622–1625. 131

[221] S. Yoon, J. Feng, and A. K. Jain, "Altered fingerprints: Analysis and detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 451–464, 2012. 131, 145

[222] F. Alonso-Fernandez, J. Fierrez, J. Ortega-Garcia, J. Gonzalez-Rodriguez, H. Fronthaler, K. Kollreider, and J. Bigun, "A comparative study of fingerprint image-quality estimation methods," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 4, pp. 734–743, 2007. 131

[223] J. Fierrez-Aguilar, Y. Chen, J. Ortega-Garcia, and A. K. Jain, "Incorporating image quality in multi-algorithm fingerprint verification," in *ICB.* Springer, 2006, pp. 213–220. 131

[224] E. Tabassi and P. Grother, "Fingerprint image quality," in *Encyclopedia of Biometrics.* Springer, 2009, pp. 482–490. 131

[225] S. Chikkerur, A. N. Cartwright, and V. Govindaraju, "Fingerprint enhancement using STFT analysis," *Pattern Recognition*, vol. 40, no. 1, pp. 198–211, 2007. 131

[226] J. Feng, J. Zhou, and A. K. Jain, "Orientation field estimation for latent fingerprint enhancement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 925–940, 2013. 131

[227] L. Hong, Y. Wan, and A. Jain, "Fingerprint image enhancement: Algorithm and performance evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 777–789, 1998. 131

[228] F. Turroni, R. Cappelli, and D. Maltoni, "Fingerprint enhancement using contextual iterative filtering," in *Biometrics (ICB), 2012 5th IAPR International Conference on.* IEEE, 2012, pp. 152–157. 131

[229] X. Yang, J. Feng, and J. Zhou, "Localized dictionaries based orientation field estimation for latent fingerprints," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 955–969, 2014. 131

[230] R. M. Bolle, R. S. Germain, R. L. Garwin, J. L. Levine, S. U. Pankanti, N. K. Ratha, and M. A. Schappert, "System and method for distortion control in live-scan inkless fingerprint images," May 16 2000, uS Patent 6,064,753. 133

[231] Y. Fujii, "Detection of fingerprint distortion by deformation of elastic film or displacement of transparent board," Feb. 9 2010, uS Patent 7,660,447. 133

[232] C. Dorai, N. K. Ratha, and R. M. Bolle, "Dynamic behavior analysis in compressed fingerprint videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 58–73, 2004. 133

[233] N. K. Ratha, K. Karu, S. Chen, and A. K. Jain, “A real-time matching system for large fingerprint databases,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 799–813, 1996. 133

[234] X. Chen, J. Tian, and X. Yang, “A new algorithm for distorted fingerprints matching based on normalized fuzzy similarity measure,” *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 767–776, 2006. 133

[235] A. M. Bazen and S. H. Gerez, “Fingerprint matching by thin-plate spline modelling of elastic deformations,” *Pattern Recognition*, vol. 36, no. 8, pp. 1859–1867, 2003. 133, 134

[236] L. R. Thebaud, “Systems and methods with identity verification by comparison and interpretation of skin patterns such as fingerprints,” Jun. 1 1999, uS Patent 5,909,501. 133

[237] Z. M. Kovacs-Vajna, “A fingerprint verification system based on triangular matching and dynamic time warping,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1266–1276, 2000. 133

[238] J. Feng, Z. Ouyang, and A. Cai, “Fingerprint matching using ridges,” *Pattern Recognition*, vol. 39, no. 11, pp. 2131–2140, 2006. 133

[239] F. L. Bookstein, “Principal warps: Thin-plate splines and the decomposition of deformations,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 567–585, 1989. 133, 136

[240] A. Ross, S. Dass, and A. Jain, “A deformable model for fingerprint matching,” *Pattern Recognition*, vol. 38, no. 1, pp. 95–103, 2005. 134

[241] A. Ross, S. C. Dass, and A. K. Jain, “Fingerprint warping using ridge curve correspondences,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 1, pp. 19–30, 2006. 134

[242] A. W. Senior and R. M. Bolle, “Improved fingerprint matching by distortion removal,” *IEICE Transactions on Information and Systems*, vol. 84, no. 7, pp. 825–832, 2001. 134

[243] D. Wan and J. Zhou, “Fingerprint recognition using model-based density map,” *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1690–1696, 2006. 134

[244] J. Feng, “Combining minutiae descriptors for fingerprint matching,” *Pattern Recognition*, vol. 41, no. 1, pp. 342–352, 2008. 134

[245] X. Si, J. Feng, and J. Zhou, “Detecting fingerprint distortion from a single image,” in *2012 IEEE International Workshop on Information Forensics and Security (WIFS)*, Dec 2012, pp. 1–6. 136, 145

[246] Neurotechnology Inc., Verifinger. http://www.neurotechnology.com. [Online]. Available: http://www.neurotechnology.com 137, 139, 160

[247] S. Novikov and O. Ushmaev, “Principal deformations of fingerprints,” in *Audio-and Video-Based Biometric Person Authentication.* Springer, 2005, pp. 229–237. 137

[248] S. Tang, Y. Fan, G. Wu, M. Kim, and D. Shen, “Rabbit: rapid alignment of brains by building intermediate templates,” *NeuroImage*, vol. 47, no. 4, pp. 1277–1287, 2009. 137

[249] D. Rueckert, A. F. Frangi, and J. A. Schnabel, “Automatic construction of 3-D statistical deformation models of the brain using nonrigid registration,” *IEEE Transactions on Medical Imaging*, vol. 22, no. 8, pp. 1014–1025, 2003. 137

[250] WVU multimodal dataset, Biometrics and Identification Innovation Center. http://biic.wvu.edu/. [Online]. Available: http://biic.wvu.edu/ 137, 156

[251] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014. 139, 152

[252] D. Maio, D. Maltoni, R. Cappelli, J. Wayman, and A. Jain, “Fvc2004: Third fingerprint verification competition,” *Biometric Authentication*, pp. 31–35, 2004. 145

[253] J. Li, J. Feng, and C.-C. J. Kuo, “Deep convolutional neural network for latent fingerprint enhancement,” *Signal Processing: Image Communication*, vol. 60, pp. 52–63, 2018. 145

[254] S. Yoon, J. Feng, and A. K. Jain, “Latent fingerprint enhancement via robust orientation field estimation,” in *Biometrics (IJCB), 2011 International Joint Conference on.* IEEE, 2011, pp. 1–8. 145

[255] ——, “On latent fingerprint enhancement,” *Biometric Technology for Human Identification VII*, vol. 7667, no. 1, p. 228, 2010. 145

[256] J. Feng, J. Zhou, and A. K. Jain, “Orientation field estimation for latent fingerprint enhancement,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 925–940, 2013. 145

[257] K. Cao and A. K. Jain, “Latent orientation field estimation via convolutional neural network,” in *Biometrics (ICB), 2015 International Conference on.* IEEE, 2015, pp. 349–356. 145

[258] A. Dabouei, H. Kazemi, S. M. Iranmanesh, J. Dawson, and N. M. Nasrabadi, “Fingerprint distortion rectification using deep convolutional neural networks,” *arXiv preprint arXiv:1801.01198*, 2018. 145

[259] X. Si, J. Feng, J. Zhou, and Y. Luo, “Detection and rectification of distorted fingerprints,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 555–568, March 2015. 145

[260] Y. Tang, F. Gao, and J. Feng, "Latent fingerprint minutia extraction using fully convolutional network," in *Biometrics (IJCB), 2017 IEEE International Joint Conference on.* IEEE, 2017, pp. 117–123. 146

[261] A. Sankaran, P. Pandey, M. Vatsa, and R. Singh, "On latent fingerprint minutiae extraction using stacked denoising sparse autoencoders," in *Biometrics (IJCB), 2014 IEEE International Joint Conference on.* IEEE, 2014, pp. 1–7. 146

[262] M. Mostofa, S. Mohamadi, J. Dawson, and N. M. Nasrabadi, "Deep GAN-based cross-spectral cross-resolution iris recognition," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 4, pp. 443–463, 2021. 146

[263] S. Mohamadi, D. A. Adjeroh, B. Behi, and H. Amindavar, "A new framework for spatial modeling and synthesis of genomic sequences," in *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* IEEE, 2020, pp. 2221–2226. 146

[264] S. Mohamadi and D. A. Adjeroh, "An information-theoretic framework for identifying age-related genes using human dermal fibroblast transcriptome data," in *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).* IEEE, 2021, pp. 2294–2300. 146

[265] M. Mostofa, F. Taherkhani, J. Dawson, and N. M. Nasrabadi, "Cross-spectral iris matching using conditional coupled GAN," in *2020 IEEE International Joint Conference on Biometrics (IJCB).* IEEE, 2020, pp. 1–9. 146

[266] M. Mostofa, S. N. Ferdous, and N. M. Nasrabadi, "A joint cross-modal super-resolution approach for vehicle detection in aerial imagery," in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II*, vol. 11413. International Society for Optics and Photonics, 2020, p. 114130O. 146

[267] U. Osahor and N. M. Nasrabadi, "Ortho-shot: Low displacement rank regularization with data augmentation for few-shot learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 2200–2209. 146

[268] ——, "Quality map fusion for adversarial learning," *arXiv preprint arXiv:2110.12338*, 2021. 146

[269] U. Osahor, H. Kazemi, A. Dabouei, and N. Nasrabadi, "Quality guided sketch-to-photo image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 820–821. 146

[270] A. Shokouhmand, C. Yang, N. D. Aranoff, E. Driggin, P. Green, and N. Tavassolian, "Mean pressure gradient prediction based on chest angular movements and heart rate variability parameters," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC).* IEEE, 2021, pp. 7170–7173. 146

[271] A. Shokouhmand, C. Antoine, B. K. Young, and N. Tavassolian, "Multi-modal framework for fetal heart rate estimation: Fusion of low-snr ecg and inertial sensors," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC).* IEEE, 2021, pp. 7166–7169. 146

[272] N. Kalantari, D. Liao, and V. G. Motti, "Characterizing the online discourse in twitter: Users' reaction to misinformation around covid-19 in twitter," in *2021 IEEE International Conference on Big Data (Big Data).* IEEE, 2021, pp. 4371–4380. 146

[273] N. Kalantari, H. Zheng, H. J. Graff, A. S. Evmenova, and V. G. Motti, "Emotion regulation for neurodiversity through wearable technology," in *2021 9th International Conference on Affective Computing and Intelligent Interaction (ACII).* IEEE, 2021, pp. 1–8. 146

[274] P. Mahapasuthanon, N. Kalantari, and V. G. Motti, "Evaluating an mhealth application: Findings on visualizing transportation and air quality," in *International Conference on Information.* Springer, 2021, pp. 301–312. 146

[275] M. Torkjazi, L. K. Farrokhvar, and B. Kamali, "Main contributing factors and the heuristic approach for assessing risk at mass gatherings," in *Operations Research Forum*, vol. 3, no. 1. Springer, 2022, pp. 1–26. 146

[276] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016. 149, 151

[277] A. Dosovitskiy and T. Brox, "Generating images with perceptual similarity metrics based on deep networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 658–666. 150

[278] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on.* IEEE, 2016, pp. 2414–2423. 150

[279] S. Motiian, Q. Jones, S. Iranmanesh, and G. Doretto, "Few-shot adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2017, pp. 6670–6680. 150

[280] S. Soleymani, A. Dabouei, H. Kazemi, J. Dawson, and N. M. Nasrabadi, "Multi-level feature abstraction from convolutional neural networks for multimodal biometric identification," in *24th International Conference on Pattern Recognition (ICPR)*, 2018. 150

[281] S. Soleymani, A. Torfi, J. Dawson, and N. M. Nasrabadi, "Generalized bilinear deep convolutional neural networks for multimodal biometric identification," in *IEEE International Conference on Image Processing (ICIP)*, 2018. 150

[282] H. Kazemi, M. Iranmanesh, A. Dabouei, S. Soleymani, and N. M. Nasrabadi, "Facial attributes guided deep sketch-to-photo synthesis," in *Computer Vision Workshops (WACVW), 2018 IEEE Winter Applications of.* IEEE, 2018, pp. 1–8. 150

[283] S. M. Iranmanesh, A. Dabouei, H. Kazemi, and N. M. Nasrabadi, "Deep cross polarimetric thermal-to-visible face recognition," *arXiv preprint arXiv:1801.01486*, 2018. 150

[284] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 539–546. 151

[285] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*. PMLR, 2020, pp. 1597–1607. 159

[286] M. D. Garris and R. M. McCabe, "Nist special database 27: Fingerprint minutiae from latent and matching tenprint images," *NIST Technical Report NISTIR*, vol. 6534, 2000. 160

[287] A. Sankaran, T. I. Dhamecha, M. Vatsa, and R. Singh, "On matching latent to latent fingerprints," in *Biometrics (IJCB), 2011 International Joint Conference On*. IEEE, 2011, pp. 1–6. 160, 162

[288] A. Sankaran, M. Vatsa, and R. Singh, "Multisensor optical and latent fingerprint database," *IEEE Access*, vol. 3, pp. 653–665, 2015. 160, 162

[289] C. Watson, M. Garris, E. Tabassi, C. Wilson, R. McCabe, S. Janet, and K. Ko, "Nist biometric image software," 2011. 160