# Security Evaluation of CNN-based Authentication System under Adversarial Attack

University of Turku
Department of Computing
Master of Science Thesis
Cybersecurity
June 2026
Arman Mahmoudifar

Supervisors:
Petri Sainio
Farshad Farahnakian

UNIVERSITY OF TURKU
Department of Computing

Arman Mahmoudifar: Security Evaluation of CNN-based Authentication System under Adversarial Attack

Master of Science Thesis, 83 p.
Cybersecurity
June 2026

---

Face verification systems are widely used in authentication, access control, surveillance, and mobile identity workflows. These systems usually follow a detect–embed–compare pipeline, where a face is detected and aligned, converted into an embedding by a deep neural network, and compared using a similarity metric and decision threshold. Although modern deep learning-based systems achieve strong clean performance, they may remain vulnerable to adversarially crafted inputs.

This thesis evaluates the clean and adversarial robustness of a pretrained face verification pipeline rather than training a new model from scratch. The implemented system uses MTCNN for face detection and alignment, InceptionResnetV1 pretrained on VGGFace2 for embedding extraction, cosine similarity for pair comparison, and a validation-selected threshold for verification decisions. The experiments follow an identity-disjoint evaluation protocol, and a clean baseline is first established using original face images.

The adversarial evaluation follows a pair-verification threat model. Two white-box gradient-based attacks are studied: the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). The attacks are generated on aligned face crops, and only the first image in each pair is perturbed. For genuine pairs, the attack aims to decrease similarity and cause false rejection, while for impostor pairs, it aims to increase similarity and cause false acceptance.

The results show that the clean system performs well under normal conditions but is substantially affected by adversarial perturbations. PGD has a stronger impact than FGSM, and impostor-pair attacks are more successful than genuine-pair attacks, which is important for security-sensitive deployments because false acceptance can have serious consequences. The thesis also evaluates lightweight test-time preprocessing defenses, including JPEG recompression, Gaussian blur, median filtering, and bit-depth reduction. These defenses provide partial robustness improvements without retraining the model. JPEG recompression at quality 50 gives the best overall trade-off, while median filtering preserves clean accuracy slightly better. Overall, the findings show that clean verification accuracy alone is not sufficient for evaluating face verification systems intended for security-critical applications.

Keywords: Face verification, adversarial robustness, adversarial attacks, FGSM, PGD, biometric authentication, lightweight preprocessing defenses

# Contents

# List of Figures

# List of Tables

# Declaration on the Use of Artificial Intelligence

During the preparation of this thesis, artificial intelligence tools, including ChatGPT, were used as supportive tools for improving grammar, academic style, clarity, and readability. They were also used to help revise sentences, structure explanations, and clarify technical concepts. The design of the study, implementation of the pipeline, experimental work, interpretation of results, and final conclusions were carried out by the author. The author takes full responsibility for the content of the thesis, and all AI-assisted outputs were critically reviewed, modified, and verified before inclusion.

# 1 Introduction

## 1.1 Background and Motivation

Face verification systems are widely used in access control, authentication, surveillance, and mobile identity workflows. Their purpose is to determine whether two facial images belong to the same person. Modern face verification systems commonly follow a detect–embed–compare pipeline. First, a face is detected and aligned. Second, the aligned face crop is mapped into a numerical embedding space by a deep neural network. Third, two embeddings are compared using a similarity metric and a decision threshold [1]–[5].

Deep learning has significantly improved the accuracy of face verification systems. Models such as FaceNet showed that faces can be represented as compact embeddings, allowing verification to be performed through distance-based or similarity-based comparison [3]. Large-scale datasets such as VGGFace2 further improved face recognition performance by providing substantial variation in pose, age, illumination, and identity [4]. As a result, deep face verification has become a practical approach for many identity-related applications.

However, the same deep neural networks that provide high accuracy may also be vulnerable to adversarial examples. Adversarial examples are inputs that have been intentionally modified using small perturbations in order to cause incorrect model behaviour. In face verification, such perturbations can be security-relevant

because the final decision depends on whether the similarity score between two face embeddings crosses a decision threshold. An adversarial attack may therefore attempt to make two images of the same identity appear dissimilar, or make two images of different identities appear similar enough to be accepted.

This creates an important security concern. A face verification system may perform well under clean conditions but fail when the input images are adversarially perturbed. For this reason, evaluating only clean verification accuracy is not sufficient. It is also necessary to examine how the system behaves under attack, how different attacks affect performance, and whether simple defensive preprocessing methods can improve robustness.

## 1.2 Research Objectives

The main objective of this thesis is to evaluate the robustness of a deep learning-based face verification pipeline under adversarial attack scenarios. The study focuses on an existing pretrained face verification pipeline rather than training a new face recognition model from scratch. To achieve this goal, the thesis has the following objectives:

- To construct and evaluate a clean face verification baseline using face detection, alignment, embedding extraction, cosine similarity, and threshold-based decision making.
- To assess the robustness of a pretrained face embedding model against adversarially crafted inputs.
- To compare the effects of two first-order adversarial attack methods, namely the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD).

- To evaluate system performance under attack using verification metrics such as accuracy, attack success rate, false acceptance rate (FAR), and false rejection rate (FRR).
- To examine whether lightweight test-time preprocessing defenses, such as JPEG recompression, Gaussian blur, median filtering, and bit-depth reduction, can improve robustness while preserving clean verification performance.

## 1.3 Research Questions

This thesis is guided by the following research questions:

1. How robust is a pretrained deep face verification pipeline against adversarially crafted face images?
2. How do different adversarial attack methods, specifically FGSM and PGD, affect the performance of the face verification system?
3. To what extent can simple test-time preprocessing defenses improve adversarial robustness, and which defense provides the best trade-off between clean performance and robustness?

## 1.4 Scope of the Study

This study focuses on the security evaluation of an existing face verification pipeline rather than the development or training of a new recognition model. The implemented pipeline uses MTCNN for face detection and alignment, InceptionResnetV1 pretrained on VGGFace2 for embedding extraction, cosine similarity for pair comparison, and a validation-selected threshold for verification decisions. This design reflects a practical face verification workflow while remaining feasible within the time and computational constraints of the thesis.

The experiments are limited to digital adversarial perturbations generated in the aligned face-crop space. The study evaluates two well-established white-box attacks, FGSM and PGD. The aim is not to evaluate all possible attack methods, but to compare a fast one-step attack with a stronger iterative attack under a consistent experimental setup.

The defense evaluation is also limited in scope. The thesis considers lightweight test-time input transformations, including JPEG recompression, Gaussian blur, median filtering, and bit-depth reduction. These methods are selected because they are simple to apply and do not require retraining the model. However, they are not treated as complete security solutions. Their limitations are discussed, especially because non-adaptive defense evaluations can overestimate robustness.

## 1.5 Contributions

The main contributions of this thesis are as follows:

- A reproducible face verification pipeline is implemented using pretrained face detection and embedding models. This avoids the need to train a face recognition model from scratch and makes the evaluation computationally feasible.
- A clean verification baseline is established using an identity-disjoint evaluation protocol and a validation-selected operating threshold.
- The robustness of the verification pipeline is evaluated under FGSM and PGD attacks across different perturbation strengths.
- The effects of adversarial attacks are analyzed separately for genuine and impostor verification pairs, providing insight into how attacks influence false rejection and false acceptance behaviour.

- Several lightweight test-time defenses are evaluated and compared in terms of their ability to recover adversarial performance while maintaining clean verification accuracy.
- The implementation code is made available through a GitHub repository to support transparency, reproducibility, and future research.

## 1.6 Thesis Structure

The remainder of this thesis is organized as follows. Chapter 2 reviews related work on deep face verification, adversarial attacks, and lightweight defense methods. Chapter 3 introduces the background concepts required to understand the thesis, including face verification, embedding-based comparison, adversarial examples, and evaluation metrics. Chapter 4 presents the methodology, including the dataset preparation, verification pipeline, attack generation process, and defense evaluation procedure. Chapter 5 reports the experimental results under clean, adversarial, and defended conditions. Chapter 6 discusses the meaning of the results, the implications for face verification robustness, and the limitations of the study. Finally, Chapter 7 concludes the thesis and presents directions for future work.

# 2 Literature Review

## 2.1 Overview

Deep face verification has become a dominant approach for determining whether two face images belong to the same identity. Modern systems typically rely on deep neural networks to extract compact and discriminative face representations, which are then compared using a distance or similarity metric. Recent reviews emphasize that although deep face verification systems achieve high accuracy under clean conditions, they also inherit the adversarial vulnerabilities of deep neural networks [1]. This is important because face verification is often used in security-sensitive applications, where small manipulations to the input image may lead to incorrect accept or reject decisions.

A typical face verification pipeline consists of several stages. First, a face is detected and aligned so that the input is normalized before recognition. Second, the aligned face crop is passed through a deep embedding model that maps the face image into a numerical feature space. Third, the resulting embeddings are compared using a similarity or distance metric, and a threshold is applied to decide whether the image pair should be accepted as genuine or rejected as an impostor pair [1]–[5]. The literature reviewed in this chapter therefore covers three main areas: deep face verification pipelines, adversarial attacks against deep neural networks and face recognition systems, and lightweight defenses based on input transformations.

## 2.2 Deep Face Verification Systems

Face verification differs from face identification. In face identification, the system attempts to determine which identity from a gallery corresponds to a given face image. In face verification, the system compares two face images and decides whether they belong to the same person. This makes verification naturally suitable for a pair-based evaluation setting, where genuine pairs consist of two images from the same identity and impostor pairs consist of images from different identities.

Deep learning-based face verification systems commonly use an embedding-based design. Instead of directly classifying every possible identity at test time, a neural network maps each face image into a feature vector. Verification is then performed by comparing two feature vectors. FaceNet is a key example of this approach. Schroff et al. showed that a deep network can learn a compact embedding space in which images of the same identity are close together and images of different identities are farther apart [3]. This allows verification to be performed using a simple distance or similarity threshold.

The threshold-based nature of face verification is particularly relevant for adversarial robustness evaluation. A small perturbation does not need to completely change the visual appearance of a face in order to affect the system. It only needs to move the similarity score across the decision threshold. Therefore, robustness in face verification depends not only on the embedding model, but also on the decision rule, the selected operating threshold, and the distribution of genuine and impostor scores.

## 2.3 Face Detection, Alignment, and Embedding Models

Before a face image can be embedded, most face recognition pipelines apply detection and alignment. This stage reduces irrelevant variation caused by face location, scale, and pose. MTCNN is one of the best-known lightweight cascaded methods for face detection and alignment. Zhang et al. proposed a three-stage cascaded architecture that jointly performs face detection and facial landmark localization, improving robustness under unconstrained conditions [2]. Because it provides both bounding boxes and landmarks, MTCNN has been widely used as a front end for face recognition pipelines.

After detection and alignment, the cropped face image is passed to an embedding model. Large-scale training datasets have played an important role in improving the quality of deep face representations. VGGFace2, introduced by Cao et al., contains large variation in pose, age, illumination, ethnicity, and imaging conditions, making it suitable for training robust face recognition models [4]. Models trained on such datasets can learn embeddings that generalize better across different visual conditions.

The present thesis does not train a face recognition model from scratch. Instead, it uses the `facenet-pytorch` implementation of MTCNN and InceptionResnetV1 pretrained on VGGFace2 [4], [5]. This implementation provides a practical and reproducible basis for constructing a complete verification pipeline. In this thesis, the embedding model outputs 512-dimensional embeddings for aligned $160 \times 160$ face crops. The use of a pretrained model allows the study to focus on robustness evaluation rather than model training.

## 2.4 Adversarial Examples and First-Order Attacks

The adversarial-learning literature has shown that high-performing deep neural networks can be vulnerable to small input perturbations. These perturbations are often designed to be visually minor while causing the model to make an incorrect prediction. Goodfellow et al. introduced the Fast Gradient Sign Method (FGSM), which generates an adversarial example by adding a one-step perturbation in the sign direction of the input gradient [6]. FGSM is computationally efficient and is commonly used as a basic first-order attack.

Projected Gradient Descent (PGD) extends this idea by applying multiple gradient-based update steps. Madry et al. formalized PGD as a strong iterative first-order attack in which the adversarial input is repeatedly updated and projected back into a bounded perturbation set [7]. Because PGD performs several optimization steps rather than a single step, it is generally considered a stronger baseline than FGSM for evaluating adversarial robustness [7].

In the context of face verification, gradient-based attacks can be used to manipulate the similarity between two face embeddings. For a genuine pair, an attack may try to reduce the similarity score so that two images of the same identity are incorrectly rejected. For an impostor pair, an attack may try to increase the similarity score so that two images of different identities are incorrectly accepted. This pair-based setting makes adversarial evaluation different from standard image classification, where the goal is usually to change the predicted class label of a single image.

## 2.5 Adversarial Robustness in Face Recognition

Recent research has increasingly examined adversarial attacks in the context of face recognition and face verification [8]–[10]. This is important because face recog-

nition systems are often deployed in authentication, access control, and identity-management scenarios. In such settings, an adversarially manipulated image could have security consequences if it causes an impostor to be accepted or a genuine user to be rejected [8], [10].

A key challenge in evaluating adversarial robustness for face verification is that the final decision depends on the full pipeline [9], [10]. The detector and alignment stage may affect the input passed to the embedding model. The embedding model determines the representation of each face crop. The similarity metric produces a numerical score, and the threshold converts this score into an accept or reject decision. Therefore, evaluating only the embedding model may not fully represent the behaviour of a complete verification system.

Another important issue is the operating threshold. A face verification system can appear more or less robust depending on how the threshold is selected. For this reason, it is important to select the threshold on validation data and keep it fixed during clean, adversarial, and defense evaluations. This makes the comparison more consistent and avoids adjusting the decision rule after observing adversarial results.

## 2.6 Lightweight Input-Transformation Defenses

The defense literature includes both train-time and test-time approaches. Train-time defenses usually require modifying the learning process, for example by including adversarial examples during training. However, this may not be practical when using a pretrained model or when retraining is computationally expensive. Test-time input transformations offer a simpler alternative because they can be applied before inference without changing the model parameters.

Feature Squeezing investigated techniques such as bit-depth reduction and spatial smoothing as inexpensive methods for reducing the adversarial degrees of freedom available to an attacker [11]. Similarly, Guo et al. studied input transformations

such as JPEG compression and bit-depth reduction as practical preprocessing defenses against adversarial perturbations [12]. These methods are attractive because they are simple, lightweight, and easy to apply to an existing pipeline.

However, later work has shown that such defenses must be interpreted carefully. Athalye et al. demonstrated that many defenses that appear effective against non-adaptive attacks can fail under adaptive evaluation, especially when they rely on non-differentiability, randomness, or gradient masking [13]. Therefore, lightweight input transformations should not be treated as complete security solutions. Instead, they are better understood as simple test-time robustness measures whose limitations must be clearly acknowledged.

## 2.7 Research Gap and Position of This Thesis

The reviewed literature shows that deep face verification systems are powerful but potentially vulnerable to adversarial perturbations. Previous studies have established the effectiveness of embedding-based face verification, the strength of first-order adversarial attacks such as FGSM and PGD, and the possible use of lightweight input transformations as simple defenses. However, there remains a need for complete and transparent evaluations of face verification pipelines under fixed operating conditions.

In particular, many adversarial-learning studies focus on image classification, while face verification has a different decision structure based on pairwise similarity and thresholding. In addition, some evaluations focus mainly on model-level behaviour rather than the complete verification pipeline, including detection, alignment, embedding extraction, similarity scoring, and threshold-based decision making. Lightweight defenses also require careful evaluation, because apparent robustness gains may not hold against adaptive attacks.

This thesis addresses this gap by evaluating a complete face verification pipeline

under clean and adversarial conditions. The system uses MTCNN for face detection and alignment, InceptionResnetV1 pretrained on VGGFace2 for embedding extraction, cosine similarity for pair comparison, and a validation-selected threshold for final verification decisions. The study evaluates both FGSM and PGD attacks and compares their effects on genuine and impostor pairs. It also examines several lightweight test-time input transformations, while acknowledging the limitations of non-adaptive defense evaluation.

## 2.8 Summary

This chapter reviewed the main literature supporting the thesis. Deep face verification relies on detecting and aligning faces, extracting embeddings, comparing similarity scores, and applying a decision threshold. FaceNet-style embedding models and large-scale datasets such as VGGFace2 provide the foundation for modern verification systems. At the same time, adversarial-learning research has shown that deep neural networks can be vulnerable to small perturbations, with PGD generally serving as a stronger iterative attack than FGSM. Finally, lightweight defenses such as JPEG compression, bit-depth reduction, and spatial smoothing may improve robustness in some cases, but they must be interpreted cautiously because non-adaptive evaluations can overestimate their effectiveness.

Table 2.1: Core literature that motivated the face verification and robustness evaluation pipeline

| **Topic** | **Key idea used in this thesis** | **Reference** |
|---|---|---|
| Face verification overview | Recent survey of adversarial risks in deep face verification | [1] |
| Face detection and alignment | MTCNN cascade for joint detection and landmark localization | [2] |
| Face embeddings | FaceNet-style verification using learned embeddings and threshold-based comparison | [3] |
| Pretraining dataset | VGGFace2 as a large-scale face dataset with variation in pose, age, illumination, and identity | [4] |
| Implementation | `facenet-pytorch` implementation of MTCNN and InceptionResnetV1 | [5] |
| Fast attack | FGSM as a one-step gradient-based adversarial attack | [6] |
| Strong iterative attack | PGD as an iterative first-order adversarial baseline | [7] |
| Lightweight defenses | Feature squeezing and input transformations such as smoothing, JPEG compression, and bit-depth reduction | [11], [12] |
| Adversarial face recognition | Attacks and robustness studies specifically targeting face recognition and verification systems | [8]–[10], [14] |
| Robustness evaluation | Reliable adversarial robustness evaluation requires strong attacks and careful interpretation of reported robustness | [15], [16] |
| Defense caveat | Adaptive attacks can expose weaknesses in defenses based on gradient masking or non-differentiability | [13] |

# 3 Background and Foundations

## 3.1 Overview

This chapter presents the technical background and foundational concepts required to understand the design, implementation, and evaluation of the face verification system studied in this thesis. While the previous chapter reviewed related work, this chapter focuses on the main concepts, terminology, and methods that are used throughout the experimental chapters. These foundations include face recognition and face verification, deep learning for visual representation, face detection and alignment, embedding-based comparison, similarity scoring, decision thresholds, evaluation metrics, adversarial attacks, and lightweight input transformation defenses.

The chapter begins by distinguishing face recognition, face identification, and face verification. This distinction is important because the system evaluated in this thesis is formulated as a verification task, where two face images are compared and the system decides whether they belong to the same identity. The chapter then introduces the role of convolutional neural networks in modern face recognition systems and explains how deep models transform face images into compact numerical representations called embeddings. These embeddings are later compared using cosine similarity in order to make a verification decision.

The chapter also explains the role of face detection and alignment, which are

necessary preprocessing steps before embedding extraction. In this thesis, the face verification pipeline follows a detect–embed–compare structure: the face region is first detected and aligned, the aligned face crop is then passed to a deep embedding model, and the resulting embeddings are compared using a similarity score. A decision threshold is applied to this score in order to accept or reject a pair as belonging to the same identity.

In addition to the clean face verification pipeline, this chapter introduces the main concepts of adversarial machine learning. Adversarial examples are inputs that have been intentionally modified in order to cause incorrect model predictions, often through small perturbations that may be difficult to notice visually [6], [17]. This thesis evaluates the robustness of a face verification system against two gradient-based attacks: the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). Therefore, this chapter explains the concepts of perturbation budget, white-box attack setting, input gradients, and iterative adversarial optimization.

Finally, the chapter introduces the evaluation metrics and defense concepts used later in the thesis. Metrics such as accuracy, false acceptance rate (FAR), false rejection rate (FRR), true acceptance rate (TAR), receiver operating characteristic (ROC), area under the curve (AUC), equal error rate (EER), adversarial accuracy, and attack success rate (ASR) are defined in the context of face verification. The chapter also introduces lightweight input transformation defenses, including JPEG recompression, Gaussian blurring, median filtering, and bit-depth reduction. These concepts provide the foundation for the methodology, results, and discussion presented in the following chapters.

## 3.2 Face Recognition, Identification, and Verification

Face recognition is a general term that refers to the automatic analysis of face images for recognizing or comparing human identities. In computer vision and biometric systems, face recognition can be formulated in different ways depending on the task. Two common formulations are face identification and face verification [3], [18]. Although these terms are sometimes used interchangeably in non-technical contexts, they describe different recognition problems.

Face identification is a one-to-many matching problem. In this setting, the system receives a query face image and searches for the most likely matching identity among a set of enrolled identities. For example, given one unknown face image, an identification system may compare it against a database of known individuals and return the identity with the highest similarity score. Therefore, the output of an identification system is usually an identity label or a ranked list of possible identities.

Face verification, in contrast, is a one-to-one matching problem. In this setting, the system receives two face images, or one query image and one claimed identity, and decides whether they belong to the same person. The output is therefore a binary decision: accept the pair as the same identity or reject it as different identities. This thesis focuses on face verification rather than one-to-many identification. The studied system compares two face images by extracting embeddings from both images and computing a similarity score between the two embeddings.

In a face verification task, image pairs are commonly divided into genuine pairs and impostor pairs [18], [19]. A genuine pair consists of two face images belonging to the same identity. An impostor pair consists of two face images belonging to different identities. If the system correctly accepts a genuine pair, the decision is considered correct. If it incorrectly rejects a genuine pair, this is a false rejection.

Similarly, if the system correctly rejects an impostor pair, the decision is correct. If it incorrectly accepts an impostor pair, this is a false acceptance.

The distinction between genuine and impostor pairs is central to the evaluation of a verification system. A strong face verification system should assign high similarity scores to genuine pairs and low similarity scores to impostor pairs. However, this separation is rarely perfect in practice. Images of the same person can differ because of pose, illumination, facial expression, occlusion, age, camera quality, or background conditions. At the same time, images of different people may sometimes produce similar feature representations, especially when the faces share visual similarities or when the input images are difficult to process. For this reason, verification systems usually require a decision threshold that determines whether a similarity score is high enough for acceptance [18]–[20].

In this thesis, the verification decision is based on pairwise comparison. Each face image is first processed by a face detection and alignment stage. The aligned face crop is then passed to a deep face recognition model that produces a numerical embedding. The embeddings of the two images are compared using cosine similarity, and a fixed threshold is used to decide whether the pair should be accepted as genuine or rejected as impostor. This formulation makes the task suitable for evaluating both clean verification performance and adversarial robustness, because the effect of adversarial perturbations can be measured by observing how often originally correct verification decisions become incorrect under attack.

The terminology used in this thesis follows this verification setting. The term "genuine pair" refers to a same-identity pair, while the term "impostor pair" refers to a different-identity pair. The term "accept" means that the system predicts the two images as belonging to the same identity, whereas "reject" means that the system predicts them as belonging to different identities. These definitions are used consistently in the later methodology, results, and discussion chapters.

## 3.3 Deep Learning and Convolutional Neural Networks

Deep learning is a subfield of machine learning that uses neural networks with multiple layers to learn representations from data [21]. In traditional machine learning approaches, features are often designed manually before being passed to a classifier. In contrast, deep learning models can learn hierarchical feature representations directly from raw or minimally processed input data [22]. This ability has made deep learning particularly effective in computer vision tasks such as image classification, object detection, and face recognition.

A neural network consists of interconnected computational units that transform an input into an output through a sequence of layers. Each layer applies learned parameters to the input representation and passes the result to the next layer. During training, the model adjusts its parameters in order to reduce a loss function, which measures the difference between the model output and the desired target [21]. Through this optimization process, the network learns features that are useful for the task being solved.

Convolutional Neural Networks (CNNs) are a class of neural networks designed especially for image and spatial data [23]. A CNN uses convolutional layers to apply learnable filters across local regions of an image. These filters can detect visual patterns such as edges, corners, textures, and more complex structures in deeper layers. Early layers usually capture low-level features, while deeper layers combine them into higher-level semantic representations. This hierarchical feature-learning process is one reason CNNs have become widely used in modern computer vision systems [22]–[24].

In face recognition, CNNs are commonly used as feature extractors. Rather than comparing two face images directly at the pixel level, a CNN-based face model maps

each face image to a numerical feature vector. This vector is intended to represent identity-related information while reducing the influence of irrelevant variation such as pose, illumination, expression, and background. The resulting feature vector is commonly called an embedding. Once embeddings have been extracted, face verification can be performed by comparing the embeddings of two face images using a similarity measure.

Modern face recognition systems often rely on pretrained CNN models. A pretrained model is first trained on a large dataset and can then be used as a feature extractor in another system. This is useful because training a high-quality face recognition model from the beginning requires a very large dataset and substantial computational resources. In this thesis, the face verification pipeline uses a pretrained CNN-based embedding model rather than training a new face recognition network from scratch. The model is used to extract embeddings from aligned face crops, and the verification decision is made by comparing the resulting embeddings.

In the context of this thesis, the CNN model is not treated as a general image classifier. Instead, it is used as an embedding generator. The purpose of the model is to transform each detected and aligned face image into a compact vector representation. This distinction is important because the system does not directly classify a face image into an identity label. Instead, it compares two embedding vectors and decides whether the two input images belong to the same person. Therefore, CNNs provide the representational foundation for the detect–embed–compare verification pipeline used in this work.

## 3.4 Face Detection and Alignment

Face detection and alignment are important preprocessing steps in many face recognition and face verification systems [2], [25]. Before a face image can be passed to an embedding model, the system must first locate the face region and transform it

into a consistent format. This preprocessing stage helps reduce variation caused by background, scale, position, and facial orientation. In a verification pipeline, reliable detection and alignment are especially important because errors at this stage can affect the quality of the extracted embedding and, consequently, the final similarity score.

Face detection refers to the process of locating one or more faces in an input image. A detector usually returns a bounding box that indicates the position and size of the detected face region. In face recognition systems, the detected region is then cropped so that the following recognition model focuses mainly on the face rather than the surrounding background. If face detection fails, the embedding model may receive an invalid input, and the corresponding image or image pair may need to be excluded from evaluation.

Face alignment is the process of normalizing the position and orientation of the detected face. This is commonly done using facial landmarks, such as the eyes, nose, and mouth corners. By estimating these landmarks, the system can align the face so that important facial components appear in approximately consistent positions across different images. Alignment reduces unwanted variation caused by head pose, image rotation, and scale differences [25]. This makes it easier for the embedding model to produce comparable representations for different images of the same identity.

In this thesis, face detection and alignment are performed using Multi-task Cascaded Convolutional Networks (MTCNN). MTCNN is a cascaded framework that jointly performs face detection and facial landmark localization using three stages of convolutional networks [2]. The first stage proposes candidate face regions, the second stage refines these candidates, and the final stage produces more accurate bounding boxes and facial landmarks. This combination of detection and landmark estimation makes MTCNN suitable as a preprocessing component for face recogni-

tion pipelines.

After detection and alignment, the face region is cropped and resized to match the expected input format of the embedding model. In the pipeline used in this thesis, each detected face is converted into an aligned face crop before being passed to the CNN-based embedding model. This follows a detect–embed–compare structure: first, the face is detected and aligned; second, an embedding is extracted from the aligned crop; and third, the embeddings of two images are compared using a similarity score.

The quality of the detection and alignment stage can influence both clean verification performance and adversarial robustness evaluation. If the face crop does not contain the correct face region or if the alignment is poor, the embedding may not accurately represent the identity. This can lead to incorrect verification decisions even before adversarial perturbations are considered. Therefore, the preprocessing stage is an essential part of the overall face verification system, not merely a technical detail.

In the adversarial part of this thesis, perturbations are applied after the face has already been detected and aligned. In other words, the attacks are generated in the aligned crop space rather than on the original full input image. This design allows the robustness evaluation to focus on the embedding and verification components of the system while keeping the detection stage fixed. The methodological details of this choice, including crop size and preprocessing settings, are described in Chapter 4.

## 3.5 Deep Face Embeddings

Modern face recognition systems commonly represent face images using deep feature vectors called embeddings [3], [26]. A face embedding is a numerical representation produced by a neural network, where each input face image is mapped to a point in a high-dimensional feature space. The purpose of this representation is to encode

identity-related information in a compact form. Instead of comparing two face images directly at the pixel level, a verification system compares their embeddings.

Embedding-based face recognition has become widely used because raw pixel comparison is highly sensitive to changes in pose, illumination, facial expression, image quality, and background. Two images of the same person may look different at the pixel level because they were captured under different conditions. Conversely, two images of different people may share similar low-level visual patterns. A deep embedding model aims to reduce this problem by learning features that are more useful for identity comparison than raw image values.

In an embedding space, images of the same identity are expected to be located close to each other, while images of different identities are expected to be farther apart [3], [26]. This property allows face verification to be formulated as a similarity-comparison problem. Given two aligned face images, the embedding model extracts one vector from each image. A similarity measure is then used to compare the two vectors. If the similarity is high enough, the pair is accepted as a genuine pair; otherwise, it is rejected as an impostor pair.

One influential example of embedding-based face recognition is FaceNet, which learns a mapping from face images to a compact Euclidean embedding space [3]. FaceNet demonstrated that face verification, recognition, and clustering can be performed using learned embeddings rather than manual facial features or direct pixel comparisons. This general idea is also followed in many later deep face recognition systems, where a CNN-based model is used to generate feature vectors that can be compared using a distance or similarity measure [26].

In this thesis, the embedding model is InceptionResnetV1 pretrained on the VGGFace2 dataset. InceptionResnetV1 combines Inception-style convolutional modules with residual connections, allowing the network to learn rich visual representations while maintaining efficient optimization. VGGFace2 is a large face recognition

dataset designed to include substantial variation in pose, age, illumination, ethnicity, and profession [4]. A model pretrained on such a dataset can provide useful identity-related features without requiring the face recognition network to be trained from scratch within this thesis.

For each aligned face crop, the embedding model outputs a fixed-length vector. In the implementation used in this thesis, this vector has 512 dimensions. The individual dimensions of the embedding should not be interpreted as separate human-understandable facial attributes. Instead, the embedding as a whole represents information learned by the neural network during training. The verification system then uses these embeddings as the basis for comparing two face images.

It is important to note that the embedding model does not directly make the final verification decision in this thesis. The model only transforms each face crop into a numerical representation. The final decision is made after comparing the two embeddings using a similarity score and applying a decision threshold. Therefore, the embedding model is one component of the broader detect–embed–compare pipeline. The next section explains how embedding vectors are compared and how the verification threshold is used to convert a similarity score into an accept or reject decision.

## 3.6 Similarity Scoring and Decision Thresholds

After face embeddings have been extracted from two aligned face images, the next step in a verification system is to compare these embeddings [3], [26]. The embedding model maps each face image to a numerical vector, but it does not directly decide whether two images belong to the same identity. Instead, a similarity score is computed between the two embedding vectors, and this score is then compared with a decision threshold.

In this thesis, cosine similarity is used to compare two face embeddings [3]. Given

two embedding vectors $e_a$ and $e_b$, cosine similarity measures the cosine of the angle between them. It is defined as

$$s(e_a, e_b) = \frac{e_a \cdot e_b}{\|e_a\|\|e_b\|}, \tag{3.1}$$

where $e_a \cdot e_b$ denotes the dot product of the two embeddings, and $\|e_a\|$ and $\|e_b\|$ denote their vector norms. A higher cosine similarity score indicates that the two embeddings point in a more similar direction in the embedding space. In the context of face verification, this usually means that the two face images are more likely to belong to the same identity.

The use of cosine similarity is suitable for embedding-based verification because the comparison is based on the relative direction of the vectors rather than their absolute magnitude. This is useful when embeddings are intended to encode identity-related information in a feature space. Genuine pairs are expected to produce higher similarity scores, while impostor pairs are expected to produce lower similarity scores [18], [27]. However, the two score distributions may still overlap in practice, which means that a decision threshold is required.

A decision threshold, denoted by $\tau$, converts the continuous similarity score into a binary verification decision [19], [27]. If the similarity score is greater than or equal to the threshold, the pair is accepted as a genuine pair. If the similarity score is below the threshold, the pair is rejected as an impostor pair. This decision rule can be written as

$$\hat{y} = \begin{cases} 1, & \text{if } s(e_a, e_b) \geq \tau, \\ 0, & \text{if } s(e_a, e_b) < \tau, \end{cases} \tag{3.2}$$

where $\hat{y} = 1$ indicates that the system predicts the pair as genuine, and $\hat{y} = 0$ indicates that the system predicts the pair as impostor.

The value of the threshold has a direct effect on the behaviour of the verification

system. A lower threshold makes the system more permissive, meaning that more pairs are accepted. This can reduce false rejections of genuine pairs, but it may increase false acceptances of impostor pairs. A higher threshold makes the system stricter, meaning that fewer pairs are accepted. This can reduce false acceptances, but it may increase false rejections. Therefore, threshold selection is an important part of face verification evaluation [20], [27].

In this thesis, the threshold is selected using the validation split and then kept fixed for the final test evaluation. This separation is important because the test split should be used only for final evaluation, not for selecting the operating point. Selecting the threshold on the validation split reduces the risk of tuning the system directly to the test data. Once selected, the same threshold is used for clean evaluation, adversarial evaluation, and defense evaluation.

The selected threshold represents the operating point of the verification system. An operating point defines how the system balances false acceptances and false rejections. Different applications may prefer different operating points. For example, a high-security access-control system may prefer a stricter threshold in order to reduce false acceptances, while a user-facing authentication system may also need to consider usability and avoid excessive false rejections. In this thesis, the threshold is chosen at the Equal Error Rate (EER) operating point on the validation split, where the false acceptance rate and false rejection rate are approximately equal. The concepts of false acceptance, false rejection, and EER are introduced in more detail in the next section.

Keeping the threshold fixed is especially important for adversarial robustness evaluation. If the threshold were changed after attacks were generated, the comparison between clean and adversarial performance would be less meaningful. By using the same validation-selected threshold across all test conditions, the experiments measure how adversarial perturbations and input transformations affect the

behaviour of the same verification system under a consistent decision rule.

## 3.7 Evaluation Metrics for Face Verification

Evaluation metrics are required to measure how well a face verification system distinguishes genuine pairs from impostor pairs [19], [27]. Since face verification is a binary decision problem, each comparison between two face images can be classified as either correct or incorrect. The system accepts a pair when it predicts that the two images belong to the same identity, and rejects a pair when it predicts that the two images belong to different identities. Based on this decision, four basic outcomes can be defined.

A true positive (TP) occurs when a genuine pair is correctly accepted. A false negative (FN) occurs when a genuine pair is incorrectly rejected. A true negative (TN) occurs when an impostor pair is correctly rejected. A false positive (FP) occurs when an impostor pair is incorrectly accepted. In the context of face verification, false positives are also called false acceptances, while false negatives are also called false rejections. These four outcomes form the basis for the evaluation metrics used throughout this thesis.

Accuracy measures the overall proportion of correctly classified pairs. It is defined as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3.3}$$

Accuracy provides a simple summary of verification performance. However, it should be interpreted carefully, especially when the number of genuine and impostor pairs is not balanced. For this reason, this thesis uses balanced genuine and impostor pairs in the main evaluation settings, so that accuracy is more informative as an overall measure.

In biometric verification, false acceptance and false rejection are particularly important [18]. The false acceptance rate (FAR) measures the proportion of impostor pairs that are incorrectly accepted as genuine. It is defined as

$$FAR = \frac{FP}{FP + TN}. \tag{3.4}$$

A high FAR means that the system is too permissive and may incorrectly accept different identities as the same person. In security-sensitive applications, FAR is often a critical metric because false acceptances may allow unauthorized access.

The false rejection rate (FRR) measures the proportion of genuine pairs that are incorrectly rejected as impostor pairs. It is defined as

$$FRR = \frac{FN}{FN + TP}. \tag{3.5}$$

A high FRR means that the system is too strict and may reject legitimate users. This can reduce usability, because genuine users may be denied even when they provide valid face images. Therefore, practical face verification systems often need to balance security and usability by considering both FAR and FRR.

The true acceptance rate (TAR) measures the proportion of genuine pairs that are correctly accepted. It is defined as

$$TAR = \frac{TP}{TP + FN}. \tag{3.6}$$

TAR is also related to FRR, because

$$TAR = 1 - FRR. \tag{3.7}$$

Thus, when the false rejection rate decreases, the true acceptance rate increases. TAR is useful for describing how well the system accepts genuine users at a given

operating threshold.

The relationship between false acceptances and false rejections depends on the decision threshold. A lower threshold usually increases the number of accepted pairs. This may increase TAR and reduce FRR, but it can also increase FAR. A higher threshold usually decreases the number of accepted pairs. This may reduce FAR, but it can also increase FRR. Therefore, the selected threshold determines the operating point of the verification system.

The receiver operating characteristic (ROC) curve is commonly used to evaluate a binary verification system over different decision thresholds [20]. The ROC curve shows the relationship between the true positive rate and the false positive rate as the threshold changes. In the terminology of face verification, the true positive rate corresponds to TAR, and the false positive rate corresponds to FAR. The ROC curve therefore illustrates how the system balances genuine-pair acceptance against impostor-pair false acceptance over a range of thresholds.

The area under the ROC curve (AUC) summarizes the overall separation ability of the system across thresholds. A higher AUC indicates that the system is better at assigning higher similarity scores to genuine pairs than to impostor pairs. An AUC close to 1 indicates strong separation between genuine and impostor score distributions, while an AUC close to 0.5 indicates performance close to random guessing. In this thesis, AUC is used as a threshold-independent measure of clean verification performance.

The equal error rate (EER) is another commonly used metric in biometric verification [19], [27]. EER is the point at which the false acceptance rate and false rejection rate are equal, or approximately equal. The threshold corresponding to this point can be used as an operating threshold when no application-specific preference is given between false acceptances and false rejections. In this thesis, the verification threshold is selected on the validation split at the EER operating point

and then kept fixed for evaluation on the test split.

In addition to clean verification metrics, adversarial robustness evaluation requires metrics that describe system performance under attack. Adversarial accuracy measures the accuracy of the verification system after adversarial perturbations have been applied. It is computed using the same decision rule as clean accuracy, but the input images or image pairs are adversarially modified. A decrease in adversarial accuracy indicates that the attack has reduced the reliability of the verification system.

Attack success rate (ASR) measures the proportion of originally correct decisions that become incorrect under attack. In this thesis, the adversarial attack subset is constructed from pairs that are correctly classified by the clean baseline. Therefore, ASR can be interpreted as the fraction of cleanly correct verification decisions that are successfully flipped by the adversarial attack. It is defined as

$$ASR = \frac{N_{\text{flipped}}}{N_{\text{clean-correct}}}, \tag{3.8}$$

where $N_{\text{flipped}}$ is the number of originally correct decisions that become incorrect after attack, and $N_{\text{clean-correct}}$ is the number of pairs that were correctly classified before the attack.

For genuine pairs, a successful attack causes the system to reject a pair that should be accepted. For impostor pairs, a successful attack causes the system to accept a pair that should be rejected. These two cases have different practical interpretations. Attacks on genuine pairs mainly reduce availability or usability by causing legitimate comparisons to fail. Attacks on impostor pairs are more directly security-relevant because they may cause the system to falsely accept different identities as the same person.

Together, accuracy, FAR, FRR, TAR, ROC, AUC, EER, adversarial accuracy, and ASR provide a comprehensive basis for evaluating the face verification system

studied in this thesis. Clean metrics describe the normal behaviour of the system, while adversarial metrics describe how its decisions change when the input is intentionally perturbed. These metrics are used in the methodology, results, and discussion chapters to compare clean verification performance, attack impact, and the effect of lightweight input transformation defenses.

## 3.8 Adversarial Machine Learning

Adversarial machine learning studies the behaviour of machine learning systems under intentionally designed inputs or conditions that aim to cause incorrect model behaviour [28]. In computer vision, this often involves adversarial examples: input images that have been modified by small perturbations in order to mislead a model. These perturbations may be difficult for humans to notice, but they can cause a neural network to produce an incorrect output [6], [17].

An adversarial example can be understood as a modified version of a clean input. Let $x$ denote a clean input image and $x_{adv}$ denote the corresponding adversarial image. The adversarial image is produced by adding a perturbation $\delta$ to the clean image:

$$x_{adv} = x + \delta. \tag{3.9}$$

The perturbation $\delta$ is usually constrained so that the adversarial image remains visually similar to the original image. In image-based attacks, this constraint is often expressed using a norm bound, such as an $L_\infty$ bound. The purpose of the constraint is to limit how much each pixel value can be changed. Therefore, the attack tries to find a perturbation that is small enough to preserve visual similarity but effective enough to change the model's decision.

In classification tasks, adversarial attacks often aim to make the model assign an

incorrect class label to an input image. In face verification, the objective is slightly different because the system compares two images rather than classifying one image into a fixed identity label. An attack may attempt to make a genuine pair appear less similar so that the system rejects it, or make an impostor pair appear more similar so that the system accepts it. Therefore, adversarial attacks against face verification can affect both usability and security.

The strength and interpretation of an adversarial attack depend on the attacker's knowledge of the system [28]. In a white-box attack setting, the attacker is assumed to have access to important details of the model, such as its architecture, parameters, loss function, and gradients. This allows the attacker to compute perturbations directly using gradient information. In a black-box setting, the attacker has limited or no access to the internal model details and may rely only on model outputs, transferability from another model, or repeated queries to the target system.

This distinction is important because white-box attacks often represent a strong evaluation setting. If a model is vulnerable even when the attacker is restricted to small perturbations, this indicates that the model may have limited robustness under adversarial conditions. However, white-box robustness does not necessarily describe all real-world attack scenarios. Practical systems may involve additional components such as image acquisition, face detection, preprocessing, network communication, rate limiting, and access-control mechanisms. For this reason, adversarial robustness evaluation should be interpreted in relation to the assumptions made about the attacker and the system.

Adversarial attacks can also be described as targeted or untargeted. In a targeted attack, the attacker tries to force the model toward a specific desired output. In an untargeted attack, the attacker only tries to cause any incorrect decision. In the context of verification, an attack can be considered successful if it changes a previously correct verification decision into an incorrect one. For a genuine pair,

this means changing an accept decision into a reject decision. For an impostor pair, this means changing a reject decision into an accept decision.

In this thesis, adversarial robustness is evaluated using gradient-based white-box attacks applied in the aligned face-crop space. This means that the face detection and alignment stage is performed first, and adversarial perturbations are then generated for the cropped face images used by the embedding model. This design focuses the robustness evaluation on the embedding and comparison stages of the face verification pipeline. The specific attacks used in this thesis, namely the Fast Gradient Sign Method and Projected Gradient Descent, are introduced in the next section.

## 3.9 Gradient-Based Adversarial Attacks

Gradient-based adversarial attacks generate perturbations by using information from the gradient of a loss function with respect to the input image. In a white-box setting, the attacker is assumed to have access to the model and can therefore compute how small changes in the input affect the model output. This gradient information can be used to modify the input in a direction that increases the loss and makes the model more likely to produce an incorrect decision.

Let $x$ denote a clean input image, $y$ denote the correct label or desired verification outcome, and $L(\theta, x, y)$ denote the loss function of a model with parameters $\theta$. The gradient of the loss with respect to the input, $\nabla_x L(\theta, x, y)$, indicates how the input image should be changed in order to increase the loss. Gradient-based attacks use this information to construct an adversarial image $x_{adv}$ that remains close to the original input but changes the model's decision.

The Fast Gradient Sign Method (FGSM) is a one-step gradient-based adversarial attack introduced by Goodfellow et al. [6]. FGSM computes the sign of the gradient of the loss with respect to the input and then moves the input image by a small amount in that direction. The attack can be written as

$$x_{adv} = \text{clip}\left(x + \epsilon \cdot \text{sign}(\nabla_x L(\theta, x, y)), -1, 1\right), \tag{3.10}$$

where $\epsilon$ controls the maximum perturbation size, sign$(\cdot)$ returns the element-wise sign of the gradient, and the clipping operation keeps the resulting adversarial image within the valid input range. In this thesis, the input range is normalized, and therefore the clipping bounds are written as $[-1, 1]$.

The parameter $\epsilon$ is often called the perturbation budget. It determines the maximum allowed change applied to the input under the chosen norm constraint. It is important to note that $\epsilon$ should not be interpreted as the percentage of image pixels that are modified. For example, $\epsilon = 0.010$ does not mean that only one percent of the image is changed. Instead, it means that the value of each pixel component can be changed by at most 0.010 in the normalized input scale, subject to the attack formulation and clipping range.

FGSM is computationally efficient because it requires only one gradient calculation. However, this also limits its strength. Since the attack takes only a single step, it may not find the most effective perturbation within the allowed perturbation budget. For this reason, stronger iterative attacks are often used to evaluate adversarial robustness more thoroughly [15].

Projected Gradient Descent (PGD) is an iterative gradient-based attack that extends the same basic idea as FGSM by applying multiple smaller steps. Instead of generating the adversarial image in one update, PGD repeatedly modifies the input in the direction of the gradient and then projects the result back into the allowed perturbation region. PGD can be written as

$$x_{t+1} = \Pi_{B_\infty(x,\epsilon)}\left[x_t + \alpha \cdot \text{sign}(\nabla_x L(\theta, x_t, y))\right], \tag{3.11}$$

where $x_t$ is the adversarial image at iteration $t$, $\alpha$ is the step size, and $\Pi_{B_\infty(x,\epsilon)}$

denotes projection onto the $L_\infty$ ball around the original input $x$ with radius $\epsilon$. The projection step ensures that the adversarial image remains within the allowed perturbation budget. In addition, the adversarial image is clipped to the valid input range after each update.

The $L_\infty$ constraint limits the maximum absolute change applied to any individual pixel component. Under this constraint, every pixel value in the adversarial image must remain within $\epsilon$ of the corresponding pixel value in the clean image. This type of constraint is commonly used in adversarial image attacks because it provides a simple way to restrict the maximum perturbation applied to each input component.

PGD is generally stronger than FGSM because it performs multiple optimization steps within the same perturbation budget. Each step can gradually adjust the input toward a more effective adversarial example. Madry et al. describe PGD as a strong first-order adversary for robustness evaluation under norm-bounded perturbations [7]. Therefore, if a model performs poorly under PGD, this indicates that the model is vulnerable to a stronger iterative white-box attack.

In the context of face verification, FGSM and PGD are applied to modify face crops in a way that changes the verification decision. For a genuine pair, the attack aims to reduce the similarity between the two embeddings so that the system rejects a pair that should be accepted. For an impostor pair, the attack aims to increase the similarity between the embeddings so that the system accepts a pair that should be rejected. In both cases, the attack is considered successful when an originally correct verification decision becomes incorrect.

In this thesis, both FGSM and PGD are evaluated using the same validation-selected verification threshold. This makes it possible to compare the effect of different attacks under a consistent decision rule. FGSM provides a one-step baseline attack, while PGD provides a stronger iterative attack. The comparison between these two methods helps show how the face verification system behaves under dif-

ferent levels of adversarial attack strength.

## 3.10 Adversarial Robustness Evaluation

Adversarial robustness evaluation measures how the performance of a machine learning system changes when its inputs are intentionally perturbed [16]. In the context of this thesis, robustness evaluation focuses on whether a face verification system can maintain correct decisions when adversarial perturbations are applied to aligned face crops. This evaluation is performed by comparing the system's clean performance with its performance under adversarial attack.

The clean baseline represents the behaviour of the verification system when no adversarial perturbation is applied. In this setting, each image pair is processed through the normal detect–embed–compare pipeline. The face images are detected and aligned, embeddings are extracted, cosine similarity is computed, and the validation-selected threshold is applied to make the final verification decision. The clean baseline is important because adversarial robustness should be interpreted relative to the system's normal verification performance.

In adversarial evaluation, the input is modified in order to change the system's decision. Let $\hat{y}_{clean}$ denote the decision made by the system on the clean input and $\hat{y}_{adv}$ denote the decision made after adversarial perturbation. An attack is successful when a previously correct clean decision becomes incorrect after the attack. Therefore, robustness evaluation measures not only whether the system performs well under normal conditions, but also whether its decisions remain stable when the input is deliberately changed.

In this thesis, adversarial evaluation is performed on an attack subset constructed from cleanly correct test pairs. A cleanly correct pair is a pair that is correctly classified by the clean verification system before any attack is applied. This design makes the attack success rate easier to interpret, because the evaluation focuses

on decisions that the system originally handled correctly. If such a decision becomes incorrect after perturbation, the change can be attributed to the effect of the adversarial attack rather than to an error that already existed in the clean baseline.

Adversarial accuracy measures the accuracy of the verification system after adversarial perturbations have been applied. It is computed using the same decision threshold and the same decision rule as in the clean evaluation. The difference is that the input images used for comparison have been modified by an attack. A lower adversarial accuracy indicates that the attack has successfully reduced the reliability of the verification system.

Attack success rate (ASR) measures the proportion of cleanly correct decisions that are flipped by an adversarial attack. It can be written as

$$ASR = \frac{N_{\text{flipped}}}{N_{\text{clean-correct}}}, \tag{3.12}$$

where $N_{\text{flipped}}$ is the number of originally correct decisions that become incorrect under attack, and $N_{\text{clean-correct}}$ is the number of pairs that were correctly classified before the attack. Since the attack subset in this thesis consists of cleanly correct pairs, adversarial accuracy and ASR are closely related: when more decisions are flipped by the attack, adversarial accuracy decreases and ASR increases.

The interpretation of attack success depends on whether the attacked pair is genuine or impostor. For a genuine pair, the clean system should accept the pair as belonging to the same identity. A successful attack causes the system to reject the pair, producing a false rejection. This type of attack mainly affects availability and usability, because a legitimate comparison fails. For an impostor pair, the clean system should reject the pair as belonging to different identities. A successful attack causes the system to accept the pair, producing a false acceptance. This case is particularly important from a security perspective, because it may allow two different identities to be incorrectly matched.

Robustness can also be studied across different perturbation budgets. An epsilon sweep evaluates the attack at multiple values of $\epsilon$. Smaller values of $\epsilon$ represent more restricted perturbations, while larger values allow stronger modifications to the input. By comparing adversarial accuracy and ASR across different $\epsilon$ values, it is possible to observe how sensitive the verification system is to increasing perturbation strength.

The same fixed verification threshold should be used across clean and adversarial evaluations. If the threshold were changed separately for each attack condition, it would be difficult to determine whether performance changes were caused by the attack or by threshold adjustment. In this thesis, the threshold is selected on the validation split and then fixed for the clean baseline, adversarial attacks, and defense experiments. This makes the comparison between different conditions more consistent.

Adversarial robustness evaluation should be interpreted in relation to the threat model, since the attacker assumptions and perturbation constraints define the meaning of robustness under attack [7]. The threat model defines the attacker's assumptions, including the attacker's knowledge of the system, the allowed perturbation size, the location where perturbations are applied, and the success criterion.

Overall, adversarial robustness evaluation extends clean verification evaluation by testing whether correct decisions remain stable under intentional perturbation. In this thesis, this is measured using adversarial accuracy, attack success rate, and performance trends across different epsilon values. These metrics provide the basis for comparing FGSM and PGD attacks and for evaluating whether lightweight input transformation defenses can reduce the impact of adversarial perturbations.

## 3.11 Lightweight Input Transformation Defenses

Adversarial attacks modify input images in order to change the behaviour of a machine learning model. One possible way to reduce the effect of such perturbations is to transform the input before it is passed to the model [12]. These methods are often called input transformation defenses or preprocessing defenses. The main idea is that some adversarial perturbations may be weakened or removed when the image is compressed, smoothed, filtered, or quantized before inference.

Input transformation defenses are attractive because they are relatively simple to apply [12], [29]. They usually do not require retraining the model or changing the model architecture. Instead, the input image is processed at test time before it is given to the embedding model. In a face verification system, this means that the aligned face crop can be transformed before the embedding is extracted. The verification pipeline then continues normally by computing the embedding, comparing it with another embedding, and applying the fixed decision threshold.

One family of input transformation defenses is often related to the idea of feature squeezing. Feature squeezing reduces unnecessary input complexity by removing or limiting details that may not be essential for the main recognition task. Xu et al. introduced feature squeezing as a method for detecting adversarial examples by comparing model predictions on original and squeezed inputs [11]. Although this thesis does not use feature squeezing as a detector, the same general idea is relevant because transformations such as bit-depth reduction and smoothing reduce the amount of fine-grained information available in the input.

JPEG recompression is one example of a lightweight transformation [12]. JPEG compression is a lossy image-compression method that removes some high-frequency image information. Since adversarial perturbations may contain small high-frequency changes, recompressing an image can sometimes reduce their effect. However, JPEG recompression can also remove useful image details, especially if the compression

quality is too low. Therefore, its effect on verification performance must be evaluated empirically.

Gaussian blurring is another simple transformation. It smooths the image by averaging neighbouring pixel values using a Gaussian kernel. This can reduce small local variations and high-frequency noise, including some adversarial perturbations. At the same time, excessive blurring can weaken identity-relevant facial details such as edges, texture, and local structure. Therefore, Gaussian blur may improve robustness in some cases but harm clean verification performance if applied too strongly.

Median filtering is a nonlinear filtering method that replaces each pixel value with the median value from a local neighbourhood. It is commonly used to reduce impulse-like noise while preserving edges better than simple averaging in some situations. In the context of adversarial robustness, median filtering may remove small irregular perturbations from the image. However, as with other transformations, it can also modify useful facial information and may not be effective against all types of attacks.

Bit-depth reduction reduces the number of possible intensity values used to represent each pixel. For example, reducing an image to a smaller number of levels forces nearby pixel values to become identical. This quantization can remove small perturbations because minor pixel-level changes may disappear after rounding. However, reducing bit depth too much can also degrade image quality and remove subtle features that the embedding model uses for identity representation.

These transformations are lightweight because they are applied directly to the input and do not require adversarial training or modification of the embedding network. This makes them practical for experimental comparison and easy to insert into an existing verification pipeline. In this thesis, JPEG recompression, Gaussian blurring, median filtering, and bit-depth reduction are evaluated as test-time input transformations. The transformed face crops are passed through the same embed-

ding model and compared using the same validation-selected threshold as in the clean and adversarial evaluations.

It is important to distinguish between non-adaptive and adaptive evaluation. In a non-adaptive evaluation, the attacker generates adversarial examples without explicitly optimizing through the defense transformation. The defense is then applied to the already generated adversarial examples. In an adaptive evaluation, the attacker is aware of the defense and designs the attack to remain effective even after the transformation. Adaptive evaluation is usually a stronger and more realistic test of a defense. Athalye et al. showed that some defenses that appear effective under weak or non-adaptive evaluation can fail when evaluated against adaptive attacks [13].

Therefore, the results of lightweight input transformation defenses should be interpreted carefully. If a transformation improves adversarial accuracy in a non-adaptive setting, this suggests that it can reduce the effect of the tested perturbations under those assumptions. However, it does not prove that the system is robust against an attacker who knows the defense and optimizes accordingly. For this reason, the defense experiments in this thesis are best understood as a lightweight robustness comparison rather than a complete security solution.

Overall, input transformation defenses provide a simple way to study whether preprocessing can reduce the impact of adversarial perturbations on a face verification system. They are easy to apply and useful for comparative analysis, but they also have limitations. Their effectiveness depends on the type of attack, the perturbation size, the transformation strength, and whether the attacker is adaptive. These considerations are important when interpreting the defense results presented later in this thesis.

## 3.12 Summary

This chapter introduced the main background concepts and technical foundations needed to understand the face verification system evaluated in this thesis. The chapter first distinguished between face recognition, face identification, and face verification. This distinction is important because the system studied in this work is formulated as a pairwise verification task, where two face images are compared and classified as either a genuine pair or an impostor pair.

The chapter then explained the role of deep learning and convolutional neural networks in modern face recognition systems. In the pipeline used in this thesis, a CNN-based embedding model is used to transform aligned face crops into numerical feature vectors. These embeddings provide the basis for comparing face images without relying on direct pixel-level comparison. The chapter also introduced face detection and alignment, which are necessary preprocessing steps before embedding extraction.

The verification decision was described as a threshold-based comparison of two face embeddings. Cosine similarity is used to measure the similarity between embedding vectors, and a fixed decision threshold is applied to decide whether a pair should be accepted or rejected. The chapter also introduced the main evaluation metrics used in this thesis, including accuracy, false acceptance rate (FAR), false rejection rate (FRR), true acceptance rate (TAR), receiver operating characteristic (ROC), area under the curve (AUC), equal error rate (EER), adversarial accuracy, and attack success rate (ASR).

Finally, the chapter presented the foundations of adversarial machine learning. It introduced adversarial examples, perturbation budgets, white-box and black-box attack settings, and the gradient-based attacks used in this thesis: the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). It also discussed adversarial robustness evaluation and lightweight input transformation defenses, in-

cluding JPEG recompression, Gaussian blurring, median filtering, and bit-depth reduction.

Overall, this chapter provides the conceptual basis for the methodology presented in the next chapter. Chapter 4 builds on these foundations by describing the dataset preparation, face verification pipeline, threshold selection, adversarial attack generation, and defense evaluation procedure used in the experimental work.

# 4 Methodology

This chapter describes the methodology used to evaluate the clean and adversarial robustness of the face verification pipeline. The methodology covers the experimental workflow, verification system, dataset preparation, attack formulation, and lightweight defense techniques. The aim is to provide a reproducible description of how the clean baseline, adversarial attacks, and defense evaluations were implemented.

## 4.1 Experimental Phases

The workflow was organized into seven phases. Phase 1 created a separate thesis workspace and phase 2 built the Conda environment and verified GPU execution. In Phase 3, the local face image dataset was reviewed, and separate training, validation, and test splits were created. The splits were identity-disjoint, meaning that the same person did not appear in more than one split. In Phase 4, the normal face verification system was tested using original, unmodified face images. This clean baseline was used as a reference for comparing the later adversarial attack results.

In Phase 5, a balanced subset of image pairs was prepared for the adversarial attack experiments. A balanced subset of image pairs means that the selected attack set contained an equal number of genuine and impostor pairs. This made the adversarial attack evaluation more balanced and easier to compare. The aligned face crops were also saved so that they could be reused later without repeating the face

alignment step. Phase 6 executed the adversarial benchmark by applying FGSM and PGD attacks to the cached aligned face crops using different values of $\epsilon$. This made it possible to evaluate how the verification system responded as the perturbation strength increased. Phase 7 evaluated lightweight preprocessing defenses, including JPEG recompression, Gaussian blur, median filtering, and bit-depth reduction, to test whether simple input transformations could reduce the effect of adversarial perturbations.

Table 4.1: Summary of the experimental workflow

| **Phase** | **Purpose** | **Main output** |
|---|---|---|
| Phase 1 | New isolated workspace | Separate folders for data, models, figures, results, and tables |
| Phase 2 | Runtime validation | PyTorch 2.4.1+cu121, CUDA 12.1, RTX 4090 Laptop GPU verified |
| Phase 3 | Dataset audit and split generation | 500 identities, 169,396 images, identity-disjoint pair protocol |
| Phase 4 | Clean verification baseline | MTCNN + InceptionResnetV1 + cosine similarity + validation threshold |
| Phase 5 | Attack-ready subset | 400 clean-correct test pairs, 787 cached face crops |
| Phase 6 | Adversarial benchmark | FGSM and PGD epsilon sweeps on the cached crop space |
| Phase 7 | Lightweight defense benchmark | JPEG, blur, median filter, and bit-depth defense comparison |

In Phase 2, runtime validation was performed to check the software environment and hardware configuration. The purpose of this step was to ensure that the workspace was capable of running the deep learning model and the later adversarial attack experiments. The Phase 2 environment check confirmed that the upgraded workspace was GPU-ready.

As shown in Table 4.2, PyTorch was installed with CUDA support, the NVIDIA RTX 4090 Laptop GPU was available, and cuDNN was enabled. A small matrix multiplication test was also run successfully to confirm that GPU computation worked correctly.

Table 4.2: Phase 2 runtime validation results captured from the executed notebook.

| **Item** | **Value** |
|---|---|
| Operating system | Windows 10 (10.0.26100) |
| Python | 3.10.20 |
| PyTorch | 2.4.1+cu121 |
| CUDA runtime | 12.1 |
| GPU | NVIDIA GeForce RTX 4090 Laptop GPU |
| cuDNN | Enabled |
| Matrix smoke test | $1024 \times 1024$ matmul in $\sim$0.0198 s |

The results in Table 4.2 confirm that the environment was correctly configured for GPU-based experiments. PyTorch was used as the deep learning framework, and CUDA support allowed PyTorch to use the NVIDIA GPU for acceleration. The RTX 4090 Laptop GPU was detected successfully, and cuDNN was enabled to provide optimized GPU operations for neural network computation. The matrix multiplication smoke test further confirmed that GPU computation was working correctly in practice. This validation was important because the later phases required repeated embedding extraction, adversarial example generation, and defense evaluation.

## 4.2 Verification System

The verification system used a conventional detect–embed–compare design. First, MTCNN was used to detect and align one face in each image. This step produced a $160 \times 160$ aligned face crop for each image. A margin of 14 pixels was added so that the crop did not cut the face too tightly and could keep useful facial regions near the edges. Post-processing was enabled to normalize the cropped face image and prepare it in the format expected by the embedding model [2], [5]. In this implementation, MTCNN was run on the CPU during face crop extraction to make the detection and alignment step more stable. The embedding model was run on the GPU because this step required more computation and could benefit from faster

processing.

Second, each cropped face image was passed to the InceptionResnetV1 model. This model was pretrained on the VGGFace2 dataset and was loaded using the facenet-pytorch library [4], [5]. For each face image, the model produced a 512-dimensional embedding vector. This vector is a numerical representation of the face and was later used to compare two face images.

Third, the two embedding vectors were compared using cosine similarity. If $e_a$ and $e_b$ are the embedding vectors of the two face images, the similarity score $s$ was computed as:

$$s = \frac{e_a \cdot e_b}{\|e_a\|\|e_b\|}. \tag{4.1}$$

Cosine similarity measures the directional similarity between two embedding vectors after normalization by their lengths. A higher value of $s$ indicates that the two embeddings are more similar and that the two face images are more likely to belong to the same identity. The pair was accepted as a match when $s \geq \tau$, and rejected when $s < \tau$.

The threshold $\tau$ was selected using the validation split at the Equal Error Rate (EER) operating point, where the false acceptance rate and false rejection rate are equal or approximately equal. This resulted in a validation threshold of approximately 0.227. After selection, the threshold was kept fixed for the clean test evaluation, adversarial attack evaluation, and defense evaluation.

In this modular pipeline, the embedding model was kept fixed and was not trained again on the local face dataset. Therefore, the train split was included to keep the experimental setup complete and consistent with standard machine learning practice. Although the train split was not used for training in the baseline experiment, it could support possible future extensions, such as model fine-tuning. In the baseline results reported in this work, the network was used only as a pretrained feature extractor.

Figure 4.1 illustrates the face verification pipeline used in this study. Each input image pair is first processed by MTCNN to detect and align a single face, producing normalized face crops of size $160 \times 160$. These cropped faces are then passed to the frozen InceptionResnetV1 model, which generates a 512-dimensional embedding vector for each image. The two embeddings are subsequently compared using cosine similarity, and the final verification decision was made by comparing the similarity score with a fixed threshold.

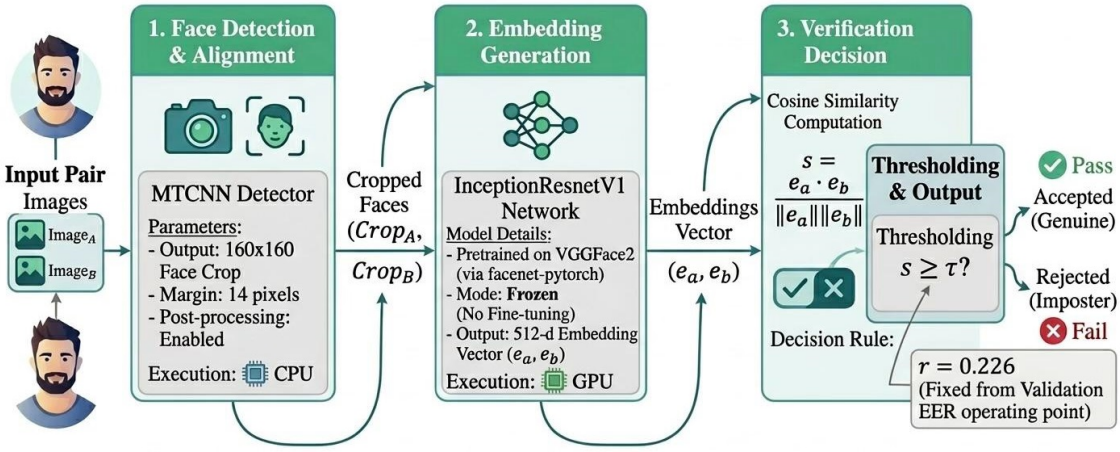


Figure 4.1: Face Verification System Architecture

The figure also makes several implementation choices explicit. In this way, the overall system can be understood as a sequence of three clearly separated stages: preprocessing, feature extraction, and similarity-based decision making. This separation is useful for later robustness analysis because it makes it easier to identify where adversarial perturbations may affect the pipeline and how defensive measures can be evaluated.

## 4.3 Dataset, Labeling, and Splits

The dataset used in this thesis was prepared from the VGGFace2 face recognition dataset [4]. VGGFace2 is a large-scale face dataset designed for face recognition

under substantial variations in pose, age, illumination, and appearance. For the experiments in this work, a subset of the dataset was used rather than the full VGGFace2 collection.

The prepared subset contained 500 identity folders and 169,396 face images in total. Each folder represented one identity and contained at least 98 images, ensuring that each identity had sufficient samples to construct verification pairs. The number of images per identity was not fixed; however, the subset remained reasonably balanced for the intended verification experiments.

Table 4.3 reports the main statistics of the prepared VGGFace2 subset. The subset included 500 identities, with a median of 325 images per identity and an average of approximately 338.8 images per identity. The variation between the minimum and maximum numbers of images shows that some identities were represented more extensively than others, but all identities satisfied the minimum image requirement used in the pair-generation procedure.

Table 4.3: Dataset audit summary from Phase 3.

| **Statistic** | **Value** |
|---|---|
| Total identities | 500 |
| Total images | 169396 |
| Minimum images per identity | 98 |
| Median images per identity | 325.0 |
| Mean images per identity | 338.792 |
| Maximum images per identity | 761 |

The median and mean values summarize the number of images per identity in two different ways. The median is the middle value after sorting the identities by the number of images. Therefore, a median of 325 means that about half of the identities had 325 images or fewer, while the other half had 325 images or more. The mean, or average, is calculated by dividing the total number of images by the number of identities. In this corpus, the mean was approximately 338.8 images per identity.

The median is less affected by extreme values than the mean. For example, the median usually does not change if the number of images is increased for the identity with the highest number of images or decreased for the identity with the lowest number of images. In contrast, the mean changes when any image count changes, because it is calculated using all identities in the corpus.

The split protocol was identity-disjoint. This means that identities, not individual images, were assigned to the train, validation, and test subsets. As a result, the same person did not appear in more than one subset. This is important for face verification because it prevents identity leakage between the training, validation, and test stages. Without this separation, the evaluation could become too optimistic because the system might be tested on identities that were already seen in another subset.

In Phase 3, the 500 identities were divided into 349 training identities, 75 validation identities, and 76 test identities. The corresponding number of images in each split is shown in Table 4.4.

Table 4.4: Identity-disjoint split sizes generated in Phase 3.

| **Split** | **No. of identities** | **No. of images** |
|---|---|---|
| Train | 349 | 117056 |
| Validation | 75 | 27310 |
| Test | 76 | 25030 |

After creating the identity-disjoint splits, pairs of face images were created separately for the train, validation, and test subsets. Each pair was used to test whether the system could decide if the two images belonged to the same person or to different people. The pair labels were binary. Genuine pairs, with label 1, contained two images from the same identity. Impostor pairs, with label 0, contained two images from different identities. Each split used a balanced number of genuine and impostor pairs, meaning that both pair types were equally represented.

Table 4.5 shows the number of identities, images, and balanced verification pairs

in each split. For example, the test split contained 3,800 verification pairs in total, consisting of 1,900 genuine pairs and 1,900 impostor pairs.

Table 4.5: Number of identities, images, and balanced verification pairs per split.

| **Split** | **No. of identities** | **No. of images** | **No. of total pairs** | **No. of genuine pairs** | **No. of impostor pairs** |
|---|---|---|---|---|---|
| Train | 349 | 117056 | 17450 | 8725 | 8725 |
| Validation | 75 | 27310 | 3750 | 1875 | 1875 |
| Test | 76 | 25030 | 3800 | 1900 | 1900 |

Although the verification pairs were generated in Phase 3, their usability was checked after clean embedding extraction in Phase 4. A pair was considered usable only when valid embeddings were available for both images in the pair. Therefore, the number of usable pairs could be slightly smaller than the number of generated pairs. Table 4.6 shows the pair coverage after clean embedding extraction.

Table 4.6: Pair coverage after clean embedding extraction in Phase 4.

| **Split** | **No. of total pairs** | **No. of usable pairs** | **Coverage Ratio** |
|---|---|---|---|
| Train | 17450 | 17403 | 0.9973 |
| Validation | 3750 | 3747 | 0.9992 |
| Test | 3800 | 3789 | 0.9971 |

The coverage ratio was high for all three splits. This shows that almost all generated verification pairs could be used in the later clean baseline evaluation. The small difference between the total and usable pairs was caused by pairs where at least one image did not produce a valid embedding.

## 4.4 Attack Formulation

The adversarial experiments followed a pair-verification threat model rather than a multiclass classification setting. In a multiclass classification setting, the model receives one image and predicts one identity label from a fixed set of classes. In this

work, however, the system received two face images and decided whether they should be accepted as a match or rejected as a match. Therefore, the attack objective was defined at the pair level, based on changing the similarity score between the two face embeddings.

The goal of a genuine-pair attack was to decrease the similarity between two images of the same person until the pair was rejected as a match. The goal of an impostor-pair attack was to increase the similarity between two images of different people until the pair was accepted as a match. Only the first image in the pair (*Image A*) was perturbed; the second image in the pair (*Image B*) was held fixed.

Phase 5 created the attack-ready subset for the adversarial experiments. First, only test pairs that were correctly classified by the clean baseline were selected. This ensured that the later attack results measured how much the adversarial perturbations changed initially correct decisions. The selected subset was balanced, containing the same number of genuine and impostor pairs.

The aligned face crops were cached once and reused during the later experiments. This avoided repeating the MTCNN face detection and alignment step for the same images. Although the subset contained 400 image pairs, it did not require 800 unique face crops because some images appeared in more than one pair. Therefore, 787 unique aligned face crops were cached.

Before the Phase 6 attack benchmark, the cached crops were scored again to check that the selected pairs were still correctly classified in the crop-space representation. This was important because the attacks were later applied directly to the aligned face crops rather than to the original full images.

Table 4.7 summarizes the final attack-ready subset created in Phase 5. The subset contained 400 usable attack pairs in total. Since the subset was balanced, it included 200 genuine pairs and 200 impostor pairs. The genuine pairs were used to evaluate whether an attack could make two images of the same person be rejected

as a match. The impostor pairs were used to evaluate whether an attack could make two images of different people be accepted as a match.

Table 4.7: Attack-ready subset created in Phase 5.

| **Phase 5 output** | **Value** |
|---|---|
| Usable attack pairs | 400 |
| Genuine pairs | 200 |
| Impostor pairs | 200 |
| Unique images represented | 787 |
| Cached crop success rate | 1.000 |

The table also shows that the 400 pairs represented 787 unique images. This number is lower than 800 because some images appeared in more than one pair. Therefore, the same cached face crop could be reused for those repeated images. The cached crop success rate of 1.000 indicates that all required aligned face crops for the attack subset were available and could be reused in the later attack experiments.

After defining the pair-verification threat model, adversarial examples were generated using two gradient-based attacks: the Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD). In this setting, $x$ represents the aligned face crop of *Image A*, and $x_{\text{adv}}$ represents its adversarially modified version. The perturbation size was controlled by $\epsilon$, which limits how much the input image is allowed to change.

FGSM generated an adversarial example using a single update step:

$$x_{\text{adv}} = \text{clip}\left(x + \epsilon \ \text{sign}(\nabla_x L), -1, 1\right). \tag{4.2}$$

In this equation, $L$ is the attack loss. The loss defines what the attack is trying to achieve. For example, for a genuine pair, the attack loss is defined so that the similarity between the two embeddings is pushed downward. For an impostor pair, the loss is defined so that the similarity is pushed upward.

The term $\nabla_x L$ is the gradient of the loss with respect to the input image $x$. This gradient shows how small changes in the input image would affect the attack

loss. In other words, it indicates which direction the pixel values should be changed in order to increase the attack objective. The sign$(\cdot)$ function keeps only the sign of each gradient value. Therefore, each pixel is changed in the positive or negative direction, instead of using the exact gradient magnitude.

The parameter $\epsilon$ controls the maximum size of the perturbation added to the image. A larger $\epsilon$ allows a stronger modification, while a smaller $\epsilon$ keeps the adversarial image closer to the original image. Finally, the clip$(\cdot)$ operation keeps the resulting adversarial image inside the valid normalized pixel range of $[-1, 1]$, so that the modified image remains a valid input for the embedding model.

The attack loss was defined according to the type of verification pair. For genuine pairs, the loss was designed to reduce the similarity between two images of the same person, so that the pair would be rejected as a match. For impostor pairs, the loss was designed to increase the similarity between two images of different people, so that the pair would be accepted as a match [6].

PGD used the same general idea as FGSM, but applied it over several smaller steps instead of one large step. The iterative update was defined as:

$$x_{t+1} = \Pi_{B_\infty(x,\epsilon)} \left[ x_t + \alpha \ \text{sign}(\nabla_x L) \right]. \tag{4.3}$$

Here, $x_t$ is the adversarial image at iteration $t$, and $x_{t+1}$ is the updated image after one PGD step. The step size is represented by $\alpha$. In this work, PGD used seven steps, and the step size was set to:

$$\alpha = \frac{\epsilon}{7}. \tag{4.4}$$

The projection operator $\Pi_{B_\infty(x,\epsilon)}$ keeps the adversarial image inside the allowed perturbation region around the original image $x$. Here, $B_\infty(x, \epsilon)$ denotes an $L_\infty$ ball centered at $x$, with radius $\epsilon$. This means that each pixel in the adversarial image

can change only up to $\epsilon$ from its original value.

This step is important because PGD applies the attack over several iterations. Although each step is small, the changes could accumulate and become larger than the allowed perturbation limit. The projection operation prevents this by clipping the adversarial image back into the allowed $L_\infty$ region after each iteration. Therefore, PGD can modify the image gradually, but the final perturbation remains bounded by $\epsilon$.

In both FGSM and PGD, the perturbation was applied in crop space rather than in the original full image. This means that the attacks were generated on the aligned $160 \times 160$ face crops after face detection and alignment, not on the complete original images before preprocessing. This distinction is important because the adversarial perturbation directly affects the input that is passed to the embedding model. However, it also means that the attack does not include the earlier face detection and alignment stage. In other words, the attack assumes that face detection and alignment have already been completed successfully. For this reason, the results should be interpreted as measuring the robustness of the embedding and similarity-based verification stages, rather than the robustness of the complete pipeline starting from the original full image.

## 4.5 Defense Techniques

Phase 7 evaluated four lightweight preprocessing defenses against the adversarial examples. These defenses were JPEG recompression, Gaussian blur, median filtering, and bit-depth reduction. A no-defense condition was also kept as a reference baseline [11], [12]. The purpose of this phase was to check whether simple image preprocessing operations could reduce the effect of adversarial perturbations before the images were passed to the embedding model.

JPEG recompression was applied with quality 50. This defense compresses the

image again and can remove some small high-frequency details from the image. Since adversarial perturbations are often small pixel-level changes, JPEG recompression may reduce part of the perturbation while keeping the main facial structure visible.

Gaussian blur was applied with radius 1.0. This operation smooths the image by averaging nearby pixel values. As a result, small local changes in the image can be weakened. This can be useful against adversarial perturbations because such perturbations often depend on small changes distributed across the image.

Median filtering was applied using a $3 \times 3$ kernel. In this method, each pixel is replaced by the median value of the pixels in its local neighborhood. Median filtering can reduce isolated pixel-level noise and small irregular changes. Therefore, it was included as another simple way to test whether local smoothing could reduce the effect of adversarial perturbations.

Bit-depth reduction was applied using 4-bit color depth. This defense reduces the number of possible pixel intensity values in the image. In other words, very small differences between pixel values are removed by mapping them to a smaller set of available values. This can reduce adversarial changes that rely on fine-grained pixel intensity differences.

The defenses were applied at test time only, and the embedding model was not retrained. For the clean defense evaluation, both images in a pair were preprocessed before embedding extraction. For the attacked condition, the defended comparison used the defended adversarial crop of *Image A* and the defended clean crop of *Image B*. This kept the evaluation consistent with the attack setup, where only *Image A* was perturbed and *Image B* was held fixed.

The original validation threshold selected in Phase 4 remained fixed during the defense evaluation. Therefore, no defense-specific threshold was selected. This made the comparison easier to interpret because the same decision threshold was used across the clean, attacked, and defended conditions. However, it also means that

the defenses were not separately calibrated for their transformed image distributions.

These defenses were lightweight because they did not require training or fine-tuning the neural network. They could be inserted before the embedding model as simple preprocessing steps. This is suitable for the modular experimental design used in this work because it isolates the effect of input transformations. However, the results should be interpreted carefully. The defense evaluation was non-adaptive, meaning that the attacker did not optimize the adversarial examples through the defense transformations. In practice, an adaptive attacker may be able to reduce or bypass the benefit of such preprocessing defenses. Therefore, the reported defense results should be understood as an initial robustness check rather than as a complete security guarantee [13].

# 5 Results

This chapter presents the experimental results of the thesis pipeline. The results are organized according to the main implementation phases, starting from workspace preparation and runtime validation, followed by dataset auditing, clean verification evaluation, adversarial attack benchmarking, and lightweight defense assessment. The chapter focuses on reporting the quantitative and qualitative findings, while their broader interpretation is discussed in Chapter 6.

## 5.1 Phase 1–2: Workspace Isolation and GPU Validation

Phase 1 achieved its engineering goal by moving all new artifacts into an isolated workspace outside the original repository. This was important because it allowed the thesis pipeline to evolve independently without modifying the original project repository. Phase 2 then confirmed that the environment could execute the pipeline on the available RTX 4090 Laptop GPU with CUDA 12.1. This matters because later phases involve large-scale embedding, repeated attack generation, and repeated defense evaluation.

## 5.2 Phase 3: Data Audit and Verification Split

Phase 3 showed that the local corpus was large enough to support the verification benchmark used in this work. All 500 identities were eligible for pair generation, with a median of 325 images per identity and a mean of 338.792 images. The split protocol produced 349 train identities, 75 validation identities, and 76 test identities. Balanced pair files were generated for each split, and the test split contained 3,800 total pairs before embedding coverage checks.

This phase is methodologically important because the split was identity-disjoint rather than image-disjoint. In a face verification study, that prevents overly optimistic evaluation caused by having the same person appear in both development and evaluation partitions. Although the baseline model was frozen, the identity-disjoint structure still made the later threshold selection and test evaluation more credible.

## 5.3 Phase 4: Clean Verification Baseline

The clean verification system performed strongly. On the validation split, the area under the ROC curve (AUC) was 0.9653 and the equal error rate (EER) was 0.0795. On the test split, the AUC increased slightly to 0.9708 and accuracy at the validation operating threshold was 0.9171. The test true accept rate (TAR) at that threshold was 0.9171, while the false accept rate (FAR) was 0.0828 and the false rejection rate (FRR) was 0.0829. These results indicate that the clean pipeline was already a strong verifier before any attack was applied.

Table 5.1: Clean verification metrics for validation and test splits.

| **Split** | **AUC** | **EER** | **EER threshold** | **Accuracy** | **TAR** | **FAR** | **FRR** | **TAR @ FAR** $10^{-2}$ | **Usable pairs** |
|---|---|---|---|---|---|---|---|---|---|
| Valid | 0.9653 | 0.0795 | 0.2268 | 0.9205 | 0.9204 | 0.0795 | 0.0796 | 0.8526 | 3747 |
| Test | 0.9708 | 0.0829 | 0.2269 | 0.9171 | 0.9171 | 0.0828 | 0.0829 | 0.8210 | 3789 |

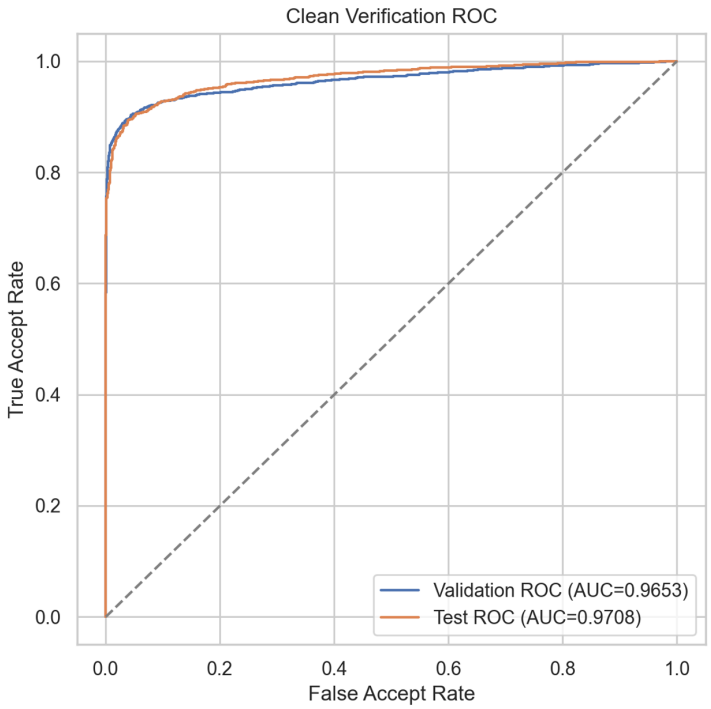


Figure 5.1: Clean verification ROC curves on validation and test splits.

Figure 5.1 shows that both ROC curves lie well above the diagonal baseline, which confirms a high degree of separability between genuine and impostor pairs in the clean setting. The validation and test curves are close, which also suggests that the threshold chosen on the validation split transferred reasonably well to the held-out test identities.

Figure 5.2 makes the decision mechanism interpretable. Genuine pairs cluster at substantially higher cosine similarity than impostor pairs, while a smaller overlap region near the threshold explains the residual false accepts and false rejects. In practical terms, the clean baseline was strong enough to justify a robustness study: if the baseline had been weak, later attack results would have been ambiguous.

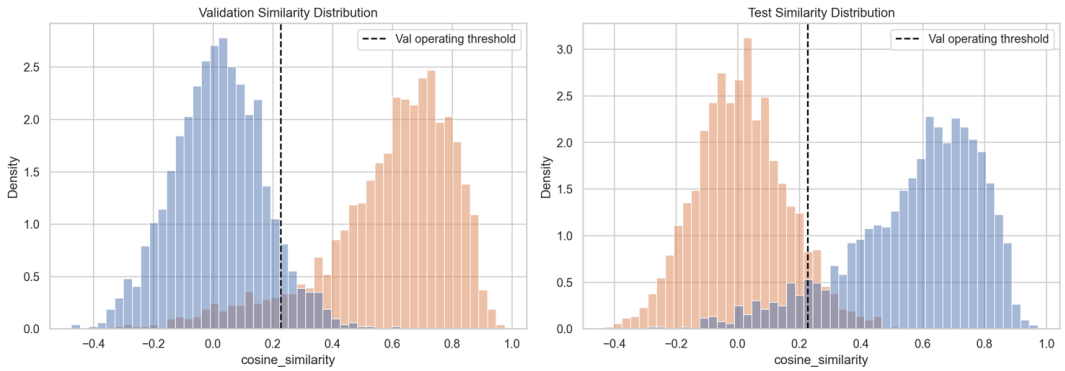


Figure 5.2: Similarity distributions for genuine and impostor pairs, with the validation operating threshold marked.

## 5.4 Phase 5: Attack-Ready Subset Construction

Phase 5 converted the clean test output into a controlled robustness benchmark. The pipeline kept only cleanly correct test pairs and then built a balanced attack subset of 400 usable pairs: 200 genuine and 200 impostor. These pairs referenced 787 unique images, and all corresponding face crops were successfully cached. This phase reduced later ambiguity because attack success in Phase 6 was measured only on pairs that were correctly classified before the attack.

That design choice is important for interpretation. If initially wrong pairs had been left in the benchmark, the measured attack success rate would have mixed clean errors with adversarial errors. By filtering to cleanly correct pairs and then rescoring the cached crop representation before attack, the benchmark focused specifically on robustness rather than baseline misclassification.

## 5.5 Phase 6: FGSM and PGD Attack Benchmark

Phase 6 produced a clear robustness result: the verifier was strongly affected by adversarial perturbations, and PGD was consistently more harmful than FGSM.

As illustrated in Table 5.2, at the mildest tested budget ($\epsilon = 0.002$), adversarial accuracy remained high at 0.9150 for FGSM and 0.9050 for PGD. However, as $\epsilon$ increased, accuracy degraded steadily. At $\epsilon = 0.010$, FGSM reduced adversarial accuracy to 0.4425 and PGD reduced it further to 0.3400. Correspondingly, attack success rose to 0.5575 for FGSM and 0.6600 for PGD.

Table 5.2: Attack benchmark summary across all FGSM and PGD epsilon levels.

| Method | Epsilon | Adv accuracy | Attack success rate | Genuine attack success rate | Impostor attack success rate | TAR adv | FAR adv |
|---|---|---|---|---|---|---|---|
| FGSM | 0.0020 | 0.9150 | 0.0850 | 0.0650 | 0.1050 | 0.9350 | 0.1050 |
| FGSM | 0.0040 | 0.7825 | 0.2175 | 0.1450 | 0.2900 | 0.8550 | 0.2900 |
| FGSM | 0.0060 | 0.6700 | 0.3300 | 0.2250 | 0.4350 | 0.7750 | 0.4350 |
| FGSM | 0.0080 | 0.5300 | 0.4700 | 0.3150 | 0.6250 | 0.6850 | 0.6250 |
| FGSM | 0.0100 | 0.4425 | 0.5575 | 0.3950 | 0.7200 | 0.6050 | 0.7200 |
| PGD | 0.0020 | 0.9050 | 0.0950 | 0.0750 | 0.1150 | 0.9250 | 0.1150 |
| PGD | 0.0040 | 0.7550 | 0.2450 | 0.1650 | 0.3250 | 0.8350 | 0.3250 |
| PGD | 0.0060 | 0.5850 | 0.4150 | 0.2750 | 0.5550 | 0.7250 | 0.5550 |
| PGD | 0.0080 | 0.4450 | 0.5550 | 0.3900 | 0.7200 | 0.6100 | 0.7200 |
| PGD | 0.0100 | 0.3400 | 0.6600 | 0.4950 | 0.8250 | 0.5050 | 0.8250 |

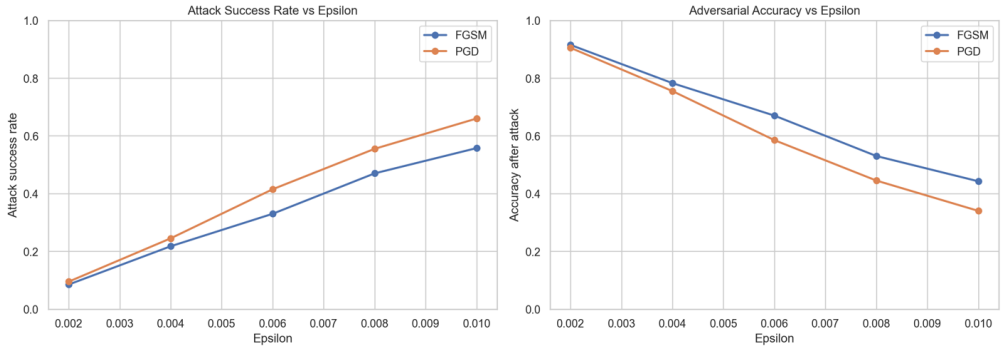


Figure 5.3: Attack success rate and adversarial accuracy as epsilon increases.

Figure 5.3 summarizes the central trend of the attack phase. Both attacks become more damaging as $\epsilon$ increases, but PGD degrades the system faster. This is exactly what would be expected from the literature: an iterative first-order attack generally explores the local adversarial region more effectively than a one-step attack [7].

Figure 5.4 also answers an important thesis question about class asymmetry. Across all tested epsilon values, impostor-pair attacks were easier than genuine-pair attacks. For example, at $\epsilon = 0.010$, FGSM achieved a genuine-pair attack success rate of 0.395 but an impostor-pair success rate of 0.720; PGD reached 0.495 on

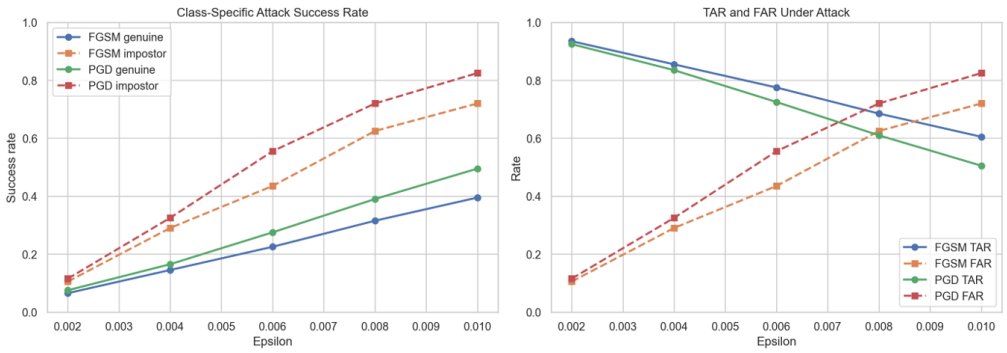


Figure 5.4: Class-specific attack success rates and verification rates under attack.

genuine pairs and 0.825 on impostor pairs. This means the system was more easily manipulated into accepting different identities than into rejecting the same identity. From a security perspective, this is important because an increased FAR under attack can be more harmful than an increased FRR in authentication scenarios.

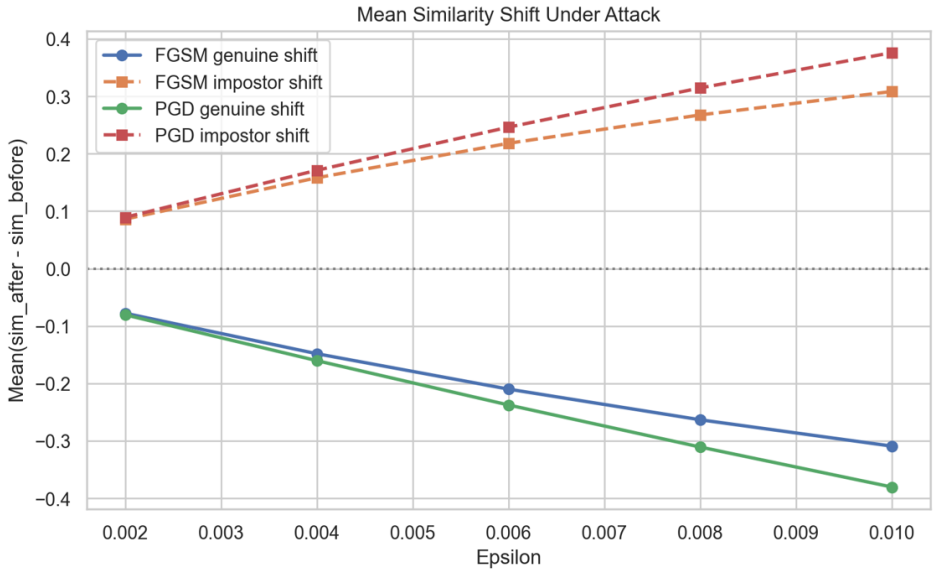


Figure 5.5: Mean similarity shift under attack for genuine and impostor pairs.

Figure 5.5 explains the mechanism behind those errors. Genuine-pair similarity shifts are negative, meaning that the attack pushes same-identity images farther apart in embedding space. Impostor-pair shifts are positive, meaning that the attack pulls different-identity images closer together. The effect grows with $\epsilon$ and is larger for PGD than FGSM, which again matches the quantitative accuracy and success-

rate results.

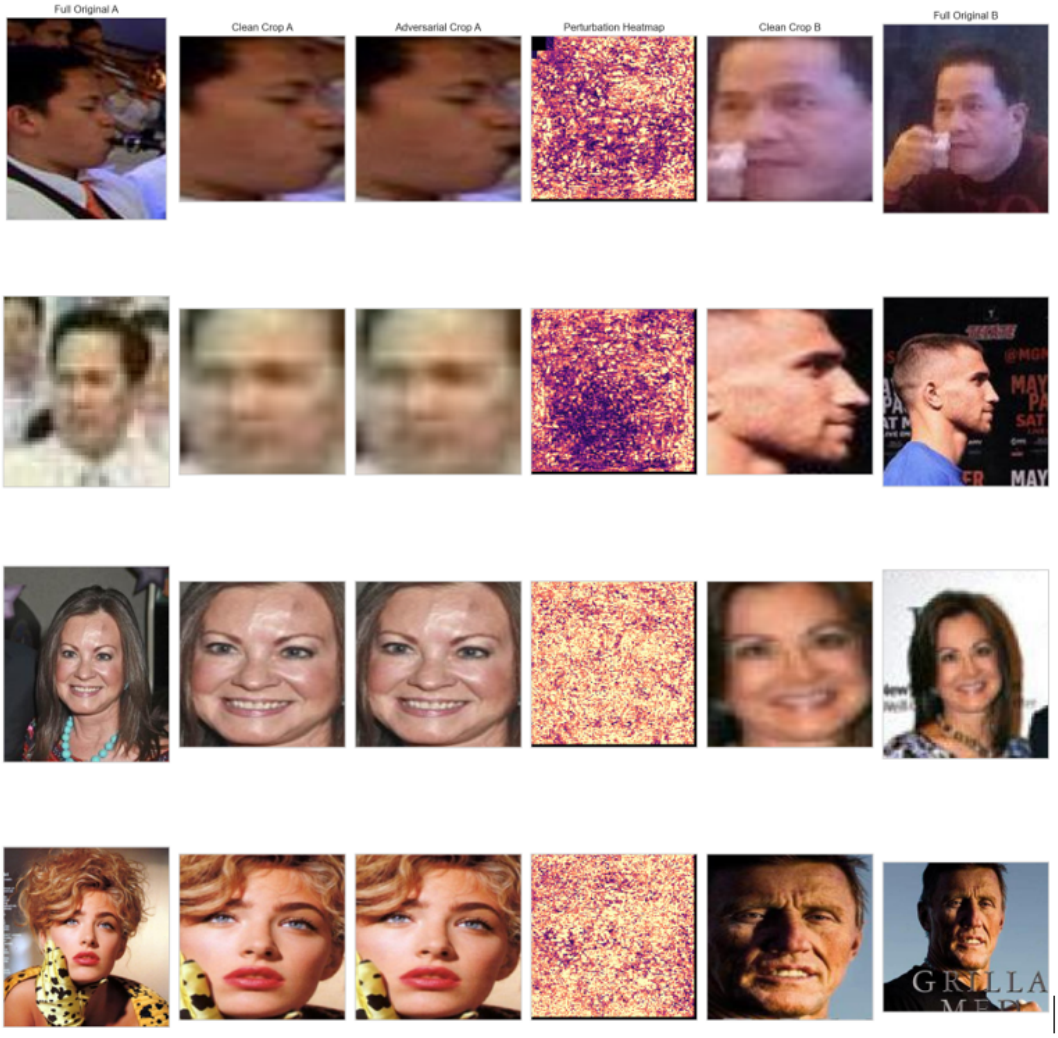


Figure 5.6: Qualitative PGD examples at $\epsilon = 0.010$, showing full images for context and crop-space perturbations for the actual attack.

Figure 5.6 provides the qualitative interpretation. The perturbation is visually subtle in the crop space even when the verification decision flips. The strongest false reject and strongest false accept rows are particularly informative: the attacked crops still appear semantically similar to a human observer, but the embedding-space similarity has moved across the threshold. The final row, which shows a strong but unsuccessful attack, is also valuable because it demonstrates that vulnerability is

substantial but not uniform across all pairs.

Table 5.3: Strongest-attack comparison at $\epsilon = 0.010$.

| Attack | Adversarial accuracy | Attack Success Rate | Adversarial TAR | Adversarial FAR | Genuine attack success rate | Impostor attack success rate |
|---|---|---|---|---|---|---|
| FGSM | 0.4425 | 0.5575 | 0.6050 | 0.7200 | 0.3950 | 0.7200 |
| PGD | 0.3400 | 0.6600 | 0.5050 | 0.8250 | 0.4950 | 0.8250 |

## 5.6 Phase 7: Lightweight Defenses

Phase 7 evaluated whether lightweight image preprocessing could reduce the effect of adversarial perturbations without requiring retraining or fine-tuning of the neural network. The tested defenses were applied before embedding extraction and were assessed in terms of both clean verification performance and adversarial robustness. The results show that simple preprocessing could partially recover robustness without severely damaging clean accuracy. The answer was positive but partial. The no-defense condition achieved 1.0000 clean accuracy on the 400-pair Phase 7 subset because the subset was intentionally constructed from pairs that were correctly classified by the clean baseline. Among the actual defenses, median filtering preserved clean accuracy best at 0.9975, while JPEG recompression remained very close at 0.9950. Bit-depth reduction and Gaussian blur both retained 0.9900 clean accuracy.

Table 5.4: Clean performance under simple preprocessing defenses

| Defense | No. of pairs | Clean accuracy | TAR | FAR | FRR | Mean genuine similarity | Mean impostor similarity |
|---|---|---|---|---|---|---|---|
| none | 400 | 1.0000 | 1.0000 | 0.0000 | 0.0000 | 0.6073 | -0.0089 |
| Median filter 3×3 | 400 | 0.9975 | 1.0000 | 0.0050 | 0.0000 | 0.6057 | -0.0080 |
| JPEG recompression (Q=50) | 400 | 0.9950 | 1.0000 | 0.0100 | 0.0000 | 0.6037 | -0.0054 |
| 4-bit color-depth reduction | 400 | 0.9900 | 0.9900 | 0.0100 | 0.0100 | 0.5975 | -0.0069 |
| Gaussian blur (r=1.0) | 400 | 0.9900 | 0.9950 | 0.0150 | 0.0050 | 0.6034 | -0.0056 |

Figure 5.7 shows that none of the tested defenses catastrophically damaged clean performance. This is an important result because a defense is of limited practical value if it only improves robustness by severely harming clean accuracy. The clean-accuracy cost here was small for all four defenses.

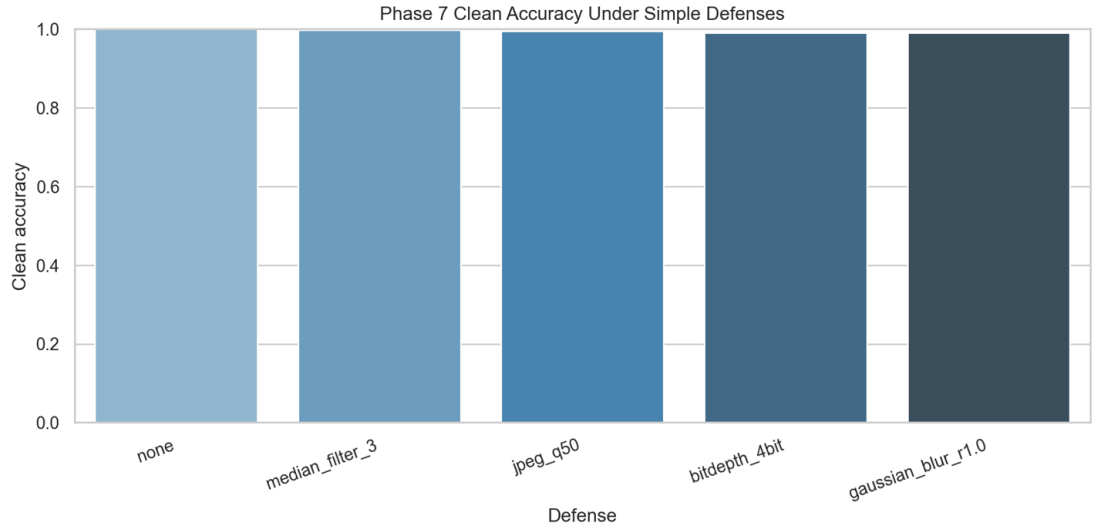


Figure 5.7: Clean accuracy under the tested preprocessing defenses.

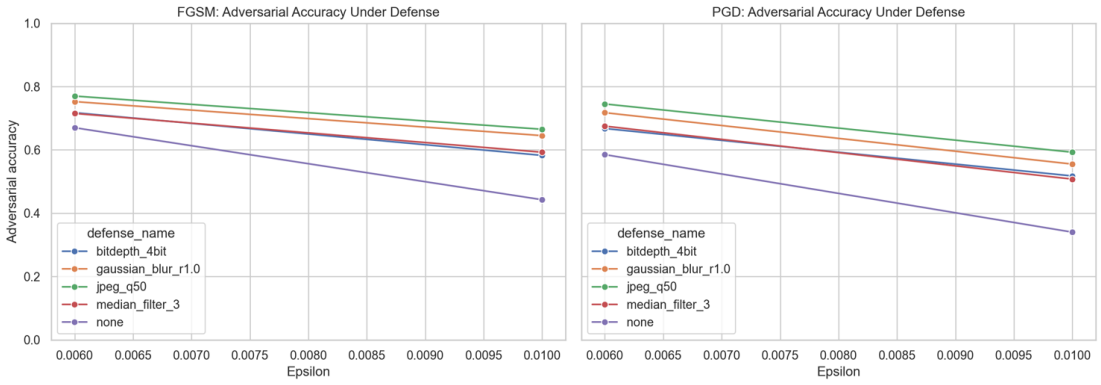


Figure 5.8: Adversarial accuracy under FGSM and PGD after defense preprocessing.

Figure 5.8 shows that all tested defenses improved adversarial accuracy relative to the no-defense baseline, but JPEG recompression was consistently strongest. At $\epsilon = 0.010$, JPEG recompression (Q=50) increased FGSM adversarial accuracy from 0.4425 to 0.6650 and PGD adversarial accuracy from 0.3400 to 0.5925. Gaussian blur ranked second, while median filtering and bit-depth reduction were weaker. This means the lightweight defenses helped substantially, but they did not restore clean-level performance.

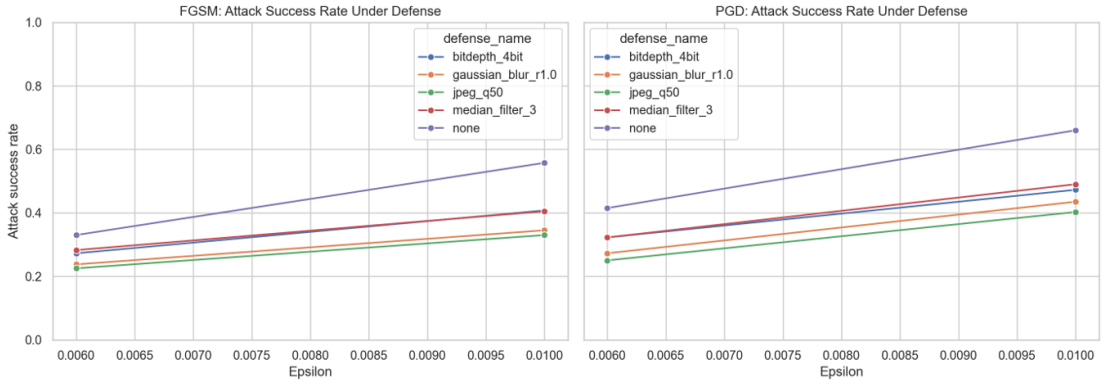


Figure 5.9: Attack success rates under defense preprocessing.

Figure 5.9 gives the same conclusion from the opposite perspective: defenses lowered attack success, and JPEG recompression (Q=50) again did so most strongly. At $\epsilon = 0.010$, FGSM attack success fell from 0.5575 with no defense to 0.3300 with JPEG recompression (Q=50), while PGD attack success fell from 0.6600 to 0.4025. The defenses therefore did not eliminate vulnerability, but they moved the system in the right direction by a meaningful margin.

Figure 5.10 makes the tradeoff easy to rank. JPEG recompression (Q=50) produced the largest gain over no defense for both adversarial accuracy and attack-success reduction, while only reducing clean accuracy from 1.0000 to 0.9950 on the Phase 7 subset. Median filtering preserved clean accuracy slightly better, but its robustness benefit was smaller. Gaussian blur was competitive, especially under PGD, but still weaker than JPEG recompression (Q=50). Bit-depth reduction helped, but

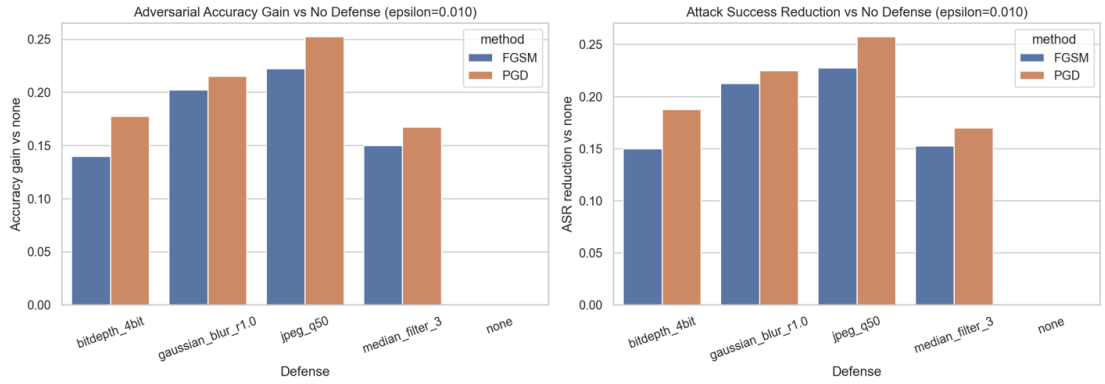


Figure 5.10: Defense gains over the no-defense baseline at the strongest tested perturbation budget.

it was the weakest of the four tested defenses at the stronger perturbation budget.

Table 5.5: Phase 7 ranking of defenses at $\epsilon = 0.010$.

| Attack | Defense | Clean Acc. | Adv. Acc. | Attack Success Rate | Adv. Acc. Gain vs. No Defense | Attack Success Reduction vs. No Defense |
|---|---|---|---|---|---|---|
| FGSM | JPEG recompression (Q=50) | 0.9950 | 0.6650 | 0.3300 | 0.2225 | 0.2275 |
| FGSM | Gaussian blur (r=1.0) | 0.9900 | 0.6450 | 0.3450 | 0.2025 | 0.2125 |
| FGSM | Median filter (3×3) | 0.9975 | 0.5925 | 0.4050 | 0.1500 | 0.1525 |
| FGSM | 4-bit color-depth reduction | 0.9900 | 0.5825 | 0.4075 | 0.1400 | 0.1500 |
| FGSM | No defense | 1.0000 | 0.4425 | 0.5575 | 0.0000 | 0.0000 |
| PGD | JPEG recompression (Q=50) | 0.9950 | 0.5925 | 0.4025 | 0.2525 | 0.2575 |
| PGD | Gaussian blur (r=1.0) | 0.9900 | 0.5550 | 0.4350 | 0.2150 | 0.2250 |
| PGD | 4-bit color-depth reduction | 0.9900 | 0.5175 | 0.4725 | 0.1775 | 0.1875 |
| PGD | Median filter (3×3) | 0.9975 | 0.5075 | 0.4900 | 0.1675 | 0.1700 |
| PGD | No defense | 1.0000 | 0.3400 | 0.6600 | 0.0000 | 0.0000 |

Overall, the defense phase produced a clear practical conclusion. If the sole priority is preserving clean performance, median filtering is the best choice among the actual defenses. If the goal is overall robustness improvement with only a small clean-accuracy penalty, JPEG recompression (Q=50) was the best option among the tested lightweight defenses.

# 6 Discussion

## 6.1 Interpretation of the Clean Baseline

The clean baseline provides the reference point for evaluating the adversarial robustness of the face verification system. Before analysing adversarial perturbations, it was necessary to confirm that the selected pipeline performed reliably on unmodified images. The baseline followed a detect–embed–compare structure: faces were detected and aligned using MTCNN, embeddings were extracted using a pretrained InceptionResnetV1 model, and verification decisions were made using cosine similarity. The verification threshold was selected on the validation split at the Equal Error Rate (EER) operating point and was kept fixed for all test, attack, and defense experiments.

The clean test results show that the baseline system achieved strong verification performance under normal conditions. On the held-out test split, the system achieved an AUC of 0.9708 and an accuracy of 0.9171. These results indicate that the embedding model was able to distinguish genuine pairs from impostor pairs effectively in the clean setting. Therefore, the baseline can be considered sufficiently reliable for evaluating how adversarial perturbations affect the verification decision.

However, strong clean performance should not be interpreted as evidence of adversarial robustness. Metrics such as clean accuracy and AUC mainly describe recognition performance under standard, non-adversarial conditions. They do not

show whether the system remains stable when the input image is intentionally modified to change the model's output. This distinction is central to the thesis: a face verification system may perform well on clean images while still being vulnerable to carefully crafted perturbations.

The use of a fixed validation-selected threshold also strengthens the interpretation of the results. Since the threshold was not adjusted after observing the test or adversarial examples, the clean, adversarial, and defense results are directly comparable. Therefore, the performance degradation observed under FGSM and PGD can be attributed to the effect of adversarial perturbations rather than to changes in the decision rule.

## 6.2 Adversarial Vulnerability of the Verification Pipeline

The adversarial attack results show that the evaluated face verification system is vulnerable to gradient-based input perturbations [14]. Although the clean baseline achieved strong performance, both FGSM and PGD caused a substantial reduction in verification accuracy. This demonstrates that small changes in the aligned face images can significantly affect the embedding representation and, consequently, the cosine similarity used for the final accept or reject decision.

At $\epsilon = 0.010$, FGSM reduced adversarial accuracy to approximately 0.4425, while PGD reduced it further to approximately 0.3400. Similarly, the attack success rate increased from approximately 0.5575 for FGSM to approximately 0.6600 for PGD. These results show that the verification decision can be manipulated by adversarial perturbations even when the clean system performs well.

The stronger effect of PGD is expected because PGD is an iterative attack, whereas FGSM is a single-step attack [6], [7]. FGSM applies one update in the

direction of the gradient, while PGD applies multiple smaller updates within the allowed perturbation budget. This allows PGD to explore the permitted perturbation space more effectively and to generate examples that are more likely to cross the verification threshold.

In the context of face verification, adversarial attacks affect the system by changing the similarity relationship between two embeddings. For genuine pairs, the attack attempts to reduce similarity so that two images of the same identity are rejected. For impostor pairs, the attack attempts to increase similarity so that two different identities are accepted. Therefore, the observed performance degradation is not only a general reduction in accuracy, but also a security-relevant change in the system's decision behaviour.

Overall, the comparison between clean and adversarial performance confirms that high clean accuracy alone is not sufficient for evaluating the security of face verification systems. A system may be reliable under normal conditions but still vulnerable when an attacker intentionally modifies the input.

## 6.3 Genuine and Impostor Pair Behaviour

The analysis of genuine and impostor pairs provides a more detailed understanding of the system's adversarial vulnerability. A genuine pair contains two images of the same identity and should be accepted by the system. An impostor pair contains two images from different identities and should be rejected. These two pair types correspond to different failure modes: false rejection for genuine pairs and false acceptance for impostor pairs.

The results indicate that adversarial attacks were generally more successful against impostor pairs than against genuine pairs. This suggests that, in the evaluated embedding space, it was easier for adversarial perturbations to increase the similarity between different identities than to decrease the similarity between im-

ages of the same identity. One possible explanation is that genuine pairs often have similarity scores with a larger margin above the threshold, making them harder to push into the rejection region. In contrast, some impostor pairs may lie closer to the threshold, so smaller perturbations can move them into the acceptance region.

This finding is important from a security perspective. False rejections mainly affect usability because a legitimate user may be denied access. False acceptances, however, are usually more serious in biometric authentication because they may allow an unauthorized identity to be accepted as genuine. Therefore, the higher success of impostor-pair attacks highlights a critical risk for face verification systems used in access control or identity verification.

The genuine–impostor analysis also shows why aggregate accuracy alone is not enough for robustness evaluation. Two systems may have similar overall adversarial accuracy but different distributions of false rejections and false acceptances. From a security viewpoint, these error types do not have equal consequences. Therefore, robustness evaluation should report not only overall accuracy and attack success rate, but also the effect of attacks on genuine and impostor pairs separately.

## 6.4 Effect of Perturbation Strength

The perturbation strength, controlled by $\epsilon$, plays an important role in adversarial robustness evaluation. In this study, $\epsilon$ defines the maximum allowed change applied to each pixel in the aligned face-crop space. A smaller $\epsilon$ restricts the attacker to weaker modifications, while a larger $\epsilon$ allows stronger changes that can more easily affect the resulting embedding.

The results show that increasing $\epsilon$ generally increased the effectiveness of both FGSM and PGD. As the perturbation budget increased, adversarial accuracy decreased and attack success rate increased. This trend is expected because a larger perturbation budget allows the attack to move the image further in the direction

that changes the model's decision.

The effect of $\epsilon$ also reflects the trade-off between attack strength and visual perceptibility. Very small perturbations may be difficult to notice but can still reduce verification performance. Larger perturbations may cause stronger attacks, but they may also become more visible and less realistic. Therefore, the security relevance of each $\epsilon$ value should be interpreted not only by its numerical effect on accuracy, but also by considering whether the perturbation remains visually plausible.

This analysis shows that robustness cannot be described by a single accuracy value. A system may appear relatively stable at a very small perturbation level but degrade rapidly as the perturbation budget increases. For this reason, evaluating several $\epsilon$ values gives a more complete picture of the system's adversarial behaviour.

## 6.5 Evaluation of Lightweight Defenses

The lightweight defense experiments examined whether simple input transformations could reduce the effect of adversarial perturbations. The evaluated methods included JPEG recompression, Gaussian blur, median filtering, and color-depth reduction. These transformations modify the input image before embedding extraction and may reduce some pixel-level perturbations.

The results show that these defenses provided only limited protection. In some cases, preprocessing may reduce high-frequency adversarial noise and partially recover verification performance. However, the same transformations can also remove useful facial details that are important for accurate embedding extraction. This creates a trade-off between suppressing perturbations and preserving identity-related information.

JPEG recompression may reduce some adversarial noise by discarding image information during compression, but it can also introduce artifacts [30]. Gaussian blur and median filtering can smooth small pixel-level variations, but excessive smoothing

may remove edges, texture, and landmark-related details. Color-depth reduction can remove small intensity changes through quantization, but it may also reduce subtle visual information needed by the model.

Among the evaluated defenses, JPEG recompression provided the best overall trade-off between clean performance and adversarial robustness in the defense evaluation setting. This suggests that compression can remove part of the perturbation pattern while preserving enough identity-related information for verification. However, this result should not be interpreted as evidence that JPEG recompression is a complete defense. Its effectiveness may depend on the attack strength, image quality, compression level, and whether the attacker is aware of the preprocessing step.

A key limitation of these defense experiments is that they were evaluated in a non-adaptive setting. The adversarial examples were not specifically optimized to bypass the preprocessing transformations. In a stronger adaptive setting, an attacker could include the defense operation in the attack process or approximate it during optimization. Therefore, the observed defense performance should be interpreted as preliminary evidence rather than proof of strong robustness [13].

Overall, the lightweight defenses should not be considered reliable standalone protections for security-critical face verification systems. They may be useful as part of a broader defense strategy, but stronger methods such as adversarial training, robust model design, anomaly detection, and deployment-level security controls are likely needed for practical systems.

## 6.6 Security Implications

The findings of this thesis have important implications for the security evaluation of face verification systems. The clean baseline showed that the system can perform well under normal conditions, but the adversarial experiments showed that this per-

formance can degrade substantially when inputs are intentionally perturbed. This confirms that clean recognition performance and adversarial robustness are different properties.

For biometric authentication, the most critical implication is the risk of false acceptance. The results showed that impostor-pair attacks were especially effective, meaning that adversarial perturbations can increase the similarity between different identities and cause the system to accept an unauthorized user. This is more serious than ordinary recognition error because it directly affects the security of the authentication process.

The results also suggest that relying only on an embedding model and a fixed similarity threshold may not be sufficient in adversarial settings. Additional mechanisms may be needed, such as liveness detection, input quality checks, anomaly detection, rate limiting, logging, and human review in high-risk cases. These mechanisms do not replace the need for robust models, but they can reduce the practical risk of successful attacks.

The findings are also relevant for production-style or API-based face verification systems. If a model is exposed through an API, attackers may submit repeated queries and observe accept or reject decisions. This can help attackers refine their attempts, especially if the system provides detailed feedback or allows unlimited requests. Therefore, adversarial robustness should be considered together with system-level protections such as access control, query limits, monitoring, and careful error-message design.

Overall, the results support the view that face verification systems should be evaluated not only as machine learning models but also as security-critical systems. A complete evaluation should consider clean accuracy, adversarial robustness, error types, defense limitations, and deployment-level risks.

## 6.7 Limitations

This study has several limitations that should be considered when interpreting the results. First, the attacks were generated in the aligned face-crop space rather than on the original raw input images. This made the evaluation more controlled and computationally efficient, but it does not fully represent an end-to-end attack against the complete image-processing pipeline. In a real deployment, adversarial images may also need to survive face detection, alignment, resizing, compression, and other preprocessing steps.

Second, the evaluation was based on one specific face verification architecture: MTCNN for detection and alignment, InceptionResnetV1 pretrained on VGGFace2 for embedding extraction, and cosine similarity for verification. Other detectors, embedding models, preprocessing pipelines, or similarity measures may show different levels of robustness.

Third, the adversarial evaluation used a fixed subset of cleanly correct test pairs. This ensured that attack success was measured only on pairs that were correctly classified before perturbation. However, this subset may not fully represent all possible genuine and impostor pair distributions in the complete test set.

Fourth, the attacks were limited to digital white-box gradient-based methods, specifically FGSM and PGD. These attacks are useful for evaluating model sensitivity, but they do not cover all possible threat scenarios. Black-box attacks, query-based attacks, transfer attacks, physical-world attacks, and presentation attacks were outside the scope of this thesis.

Fifth, the defense evaluation was limited to simple preprocessing transformations and was conducted in a non-adaptive setting. Therefore, the defense results should be interpreted cautiously. They show how the tested transformations behave under the selected conditions, but they do not prove robustness against adaptive attackers.

Finally, the study focused mainly on the machine learning component of the veri-

fication system. Although the broader motivation includes production-style and networked face verification systems, detailed API-level, network-level, and deployment-level security testing was not performed. Future work should therefore examine adversarial robustness together with practical system security mechanisms.

## 6.8 Summary of the Discussion

This chapter interpreted the experimental findings and discussed their security relevance. The clean baseline showed that the evaluated face verification system performed well under normal conditions, achieving strong AUC and accuracy on the held-out test split. However, the adversarial experiments showed that this clean performance did not guarantee robustness against intentional perturbations.

Both FGSM and PGD significantly reduced verification performance, with PGD producing stronger attacks due to its iterative optimization process. The results also showed that impostor-pair attacks were particularly important from a security perspective because they correspond to false acceptances. This finding highlights the need to evaluate adversarial robustness not only through overall accuracy, but also through separate analysis of genuine and impostor pair behaviour.

The lightweight defense experiments showed that simple preprocessing transformations can provide limited protection in some cases, but they should not be treated as reliable standalone defenses. Their effectiveness is limited, and the non-adaptive evaluation means that stronger attackers may be able to bypass them.

Overall, the findings support the main argument of this thesis: strong clean verification performance is necessary but not sufficient for security-critical face verification systems. Robustness evaluation should include adversarial attacks, analysis of error types, defense assessment, and consideration of deployment-level protections.

# 7 Conclusion and Future Work

## 7.1 Conclusion

This thesis developed and evaluated a face verification pipeline for studying clean verification performance, adversarial robustness, and lightweight defense methods. The pipeline followed a detect–embed–compare structure using MTCNN for face detection and alignment, InceptionResnetV1 pretrained on VGGFace2 for embedding extraction, cosine similarity for pair comparison, and a validation-selected threshold for verification decisions.

The study showed that a face verification system can achieve strong clean performance while still being vulnerable to adversarially crafted face images. This confirms that clean verification accuracy alone is not sufficient for evaluating systems used in security-sensitive contexts. Robustness evaluation is also necessary because adversarial perturbations can change similarity scores and cause incorrect accept or reject decisions.

The thesis also showed that lightweight preprocessing defenses can reduce the effect of adversarial perturbations to some extent, but they should not be treated as complete security solutions. Overall, the work provides a structured evaluation of the relationship between clean performance, adversarial vulnerability, and simple test-time defenses in a pretrained face verification pipeline. The direct answers to the research questions are presented in the next section.

## 7.2 Answers to the Research Questions

This section provides direct answers to the research questions presented in Chapter 1.

### RQ1: How robust is a pretrained deep face verification pipeline against adversarially crafted face images?

The evaluated pretrained deep face verification pipeline was effective under clean conditions, but it was not robust against adversarially crafted face images. The clean baseline showed strong verification performance, which made it suitable for robustness evaluation. However, adversarial perturbations substantially reduced adversarial accuracy and increased the attack success rate as the perturbation strength increased. Therefore, the system should be considered vulnerable under adversarial conditions, especially at higher epsilon values.

### RQ2: How do different adversarial attack methods, specifically FGSM and PGD, affect the performance of the face verification system?

Both FGSM and PGD reduced the performance of the face verification system, but PGD had a stronger effect across all tested epsilon values. FGSM, as a fast one-step attack, already caused a clear decrease in adversarial accuracy. PGD, as a stronger iterative attack, reduced adversarial accuracy further and produced a higher attack success rate. The results also showed that impostor-pair attacks were more successful than genuine-pair attacks, meaning that the system was more easily manipulated into falsely accepting different identities than falsely rejecting the same identity.

### RQ3: To what extent can simple test-time preprocessing defenses improve adversarial robustness, and which defense provides the best trade-off between clean performance and robustness?

Simple test-time preprocessing defenses improved adversarial robustness, but only partially. JPEG recompression, Gaussian blur, median filtering, and bit-depth reduction all improved adversarial accuracy compared with the no-defense condition while keeping clean performance relatively high. Among the tested defenses, JPEG recompression at quality 50 provided the best overall trade-off between clean performance and adversarial robustness. Median filtering preserved clean accuracy slightly better, but it did not recover as much adversarial robustness as JPEG recompression. Therefore, lightweight preprocessing defenses can reduce the effect of adversarial perturbations, but they should be treated as partial mitigation methods rather than complete security solutions.

## 7.3 Limitations

Although the study achieved its main objectives, several limitations should be considered when interpreting the results. The first limitation concerns the dataset. The evaluation dataset used in the workspace was a local corpus rather than a formally documented public benchmark release. Although the dataset was large enough to support the experimental workflow and was divided using an identity-disjoint protocol, the use of a local dataset limits direct external comparability with other published studies. Results obtained on public benchmarks are usually easier to reproduce and compare across different methods, whereas results on a local corpus mainly support internal evaluation within the scope of this thesis.

The second limitation concerns the embedding model. The face verification

system used a frozen pretrained embedding model rather than a model fine-tuned specifically for the local dataset or adversarial robustness. This design choice kept the pipeline simple, stable, and computationally feasible, which was important for the scope of a master's thesis. However, it also means that the study did not evaluate train-time robustness methods such as adversarial training, robustness-aware fine-tuning, or metric-learning-based improvements. Therefore, the defense results should be interpreted as test-time preprocessing results rather than as evidence about the maximum robustness that could be achieved by retraining or fine-tuning the recognition model.

The third limitation concerns the attack space. The adversarial attacks were applied to aligned face crops rather than to the original full-resolution input images. In the implemented pipeline, face detection and alignment were performed first, and FGSM and PGD were then applied in the crop space used by the embedding model. Therefore, the threat model is best described as a crop-space digital attack. This is useful for isolating the vulnerability of the embedding-based verification stage, but it does not fully evaluate the robustness of the complete end-to-end pipeline from raw image input to final verification decision. In a real deployment, adversarial perturbations may also affect face detection, alignment, image compression, and other preprocessing stages.

The fourth limitation concerns the defense evaluation. The lightweight defenses in Phase 7 were evaluated using a fixed-threshold, non-adaptive setting. This allowed a controlled comparison between no-defense and defended conditions, but it does not represent the strongest possible attacker. A fully adaptive attacker could optimize adversarial examples while accounting for the defense transformation. As discussed by Athalye et al., preprocessing-based defenses may appear stronger under non-adaptive evaluation than they are against adaptive attacks [13]. Therefore, the observed improvements from JPEG recompression, Gaussian blur, median filtering,

and bit-depth reduction should be interpreted as partial robustness improvements rather than complete defenses.

A final limitation is that the work focused mainly on the machine-learning component of the system. The broader deployed-system security aspects, such as API authentication, secure communication, rate limiting, logging, and network-level protection, were outside the main experimental scope. As a result, the thesis provides a focused adversarial machine-learning evaluation of the verifier, but it does not claim to provide a complete security audit of a production face verification service.

## 7.4 Future Work

Several directions can extend this work. First, future experiments should evaluate the pipeline on a formally documented public face verification benchmark. Using a public benchmark would improve external comparability and make it easier to compare the results with prior work. It would also help determine whether the findings observed in this thesis, such as the stronger effect of PGD and the greater vulnerability of impostor pairs, generalize beyond the local dataset.

Second, future work could extend the attack setting from crop-space attacks to full image-space attacks. In such a setting, adversarial perturbations would be applied before face detection and alignment. This would make it possible to evaluate the robustness of the complete end-to-end pipeline, including detection, alignment, embedding extraction, similarity scoring, and threshold-based decision-making. Such an evaluation would be closer to real-world deployment conditions, where the system receives full images rather than already aligned face crops.

Third, future research should evaluate adaptive attacks against the preprocessing defenses. In this thesis, JPEG recompression, Gaussian blur, median filtering, and bit-depth reduction were evaluated in a non-adaptive setting. A stronger evaluation would allow the attacker to optimize perturbations while taking the defense trans-

formation into account. This would provide a more realistic estimate of whether these defenses provide genuine robustness or mainly reduce the effectiveness of non-adaptive attacks.

Fourth, future work could investigate train-time defense strategies. Instead of relying only on test-time preprocessing, the embedding model could be fine-tuned using adversarial examples or trained with robustness-aware objectives. Adversarial training, robust metric learning, and data augmentation with perturbed face images could be explored to determine whether they improve robustness while preserving clean verification performance. Such methods may provide stronger protection than lightweight preprocessing alone, although they would require more computational resources and careful experimental design.

Fifth, future work could evaluate additional face recognition models and architectures. This thesis used a specific pretrained embedding model in a fixed verification pipeline. Testing multiple embedding models would show whether the observed vulnerabilities are model-specific or common across different face recognition systems. It would also allow comparison between older convolutional architectures and more recent face recognition models.

Finally, the work could be extended toward a broader deployed-system security evaluation. In addition to adversarial machine-learning robustness, future studies could examine API-level security, authentication mechanisms, rate limiting, secure communication, logging, and possible network-level weaknesses. This would connect the adversarial robustness analysis with the wider cybersecurity requirements of real-world face verification systems.

## 7.5 Final Remarks

This thesis showed that a face verification system can achieve strong clean performance while still being vulnerable to adversarial perturbations. The results

demonstrated that PGD was consistently more damaging than FGSM and that impostor-pair attacks represented a particularly important security concern. The defense evaluation showed that lightweight preprocessing methods can improve robustness, with JPEG recompression providing the strongest overall trade-off in this study. However, these defenses did not eliminate the vulnerability and should not be interpreted as complete protection against adaptive attackers.

The main contribution of the thesis is therefore not only the implementation of a working face verification and adversarial evaluation pipeline, but also the structured evidence it provides about the relationship between clean accuracy, adversarial vulnerability, and lightweight defenses. The findings support the view that adversarial robustness should be considered an essential part of evaluating face verification systems, especially when such systems are used in security-sensitive applications.

# References


[1] S. Kilany and A. Mahfouz, "A comprehensive survey of deep face verification systems adversarial attacks and defense strategies", *Scientific Reports*, vol. 15, 2025, Art. no. 30861. DOI: 10.1038/s41598-025-15753-8.

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

[3] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682.

[4] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "VGGFace2: A dataset for recognising faces across pose and age", in *Proc. 13th IEEE Int. Conf. Autom. Face Gesture Recognit.*, Xi'an, China, 2018, pp. 67–74. DOI: 10.1109/FG.2018.00020.

[5] T. Esler. "facenet-pytorch: Pretrained PyTorch face detection and facial recognition models". GitHub repository, Accessed: Jun. 14, 2026. [Online]. Available: https://github.com/timesler/facenet-pytorch.

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

[7] 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*, Vancouver, BC, Canada, 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb.

[8] 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 *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Vienna, Austria, 2016, pp. 1528–1540. DOI: 10.1145/2976749.2978392.

[9] Y. Dong, H. Su, B. Wu, Z. Li, W. Liu, T. Zhang, and J. Zhu, "Efficient decision-based black-box adversarial attacks on face recognition", in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, 2019, pp. 7706–7714. DOI: 10.1109/CVPR.2019.00790.

[10] F. Vakhshiteh, A. Nickabadi, and R. Ramachandra, "Adversarial attacks against face recognition: A comprehensive study", *IEEE Access*, vol. 9, pp. 92 735–92 756, 2021. DOI: 10.1109/ACCESS.2021.3092646.

[11] W. Xu, D. Evans, and Y. Qi, "Feature squeezing: Detecting adversarial examples in deep neural networks", in *Proc. 25th Annu. Netw. Distrib. Syst. Secur. Symp.*, San Diego, CA, USA: The Internet Society, 2018. [Online]. Available: https://www.ndss-symposium.org/wp-content/uploads/2018/02/ndss2018_03A-4_Xu_paper.pdf.

[12] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations", in *Proc. Int. Conf. Learn. Representations*, Vancouver, BC, Canada, 2018. [Online]. Available: https://openreview.net/forum?id=SyJ7ClWCb.

[13] A. Athalye, N. Carlini, and D. Wagner, "Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples", in *Proc. 35th Int.*

*Conf. Mach. Learn.*, ser. Proc. Mach. Learn. Res. Vol. 80, Stockholm, Sweden: PMLR, 2018, pp. 274–283. [Online]. Available: https://proceedings.mlr.press/v80/athalye18a.html.

[14] G. Goswami, N. Ratha, A. Agarwal, R. Singh, and M. Vatsa, "Unravelling robustness of deep learning based face recognition against adversarial attacks", in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, New Orleans, LA, USA, 2018. DOI: 10.1609/aaai.v32i1.12341.

[15] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks", in *Proc. IEEE Symp. Secur. Privacy*, San Jose, CA, USA, 2017, pp. 39–57. DOI: 10.1109/SP.2017.49.

[16] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks", in *Proc. 37th Int. Conf. Mach. Learn.*, ser. Proc. Mach. Learn. Res. Vol. 119, 2020, pp. 2206–2216. [Online]. Available: https://proceedings.mlr.press/v119/croce20b.html.

[17] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks", in *Proc. Int. Conf. Learn. Representations*, Banff, AB, Canada, 2014. [Online]. Available: https://openreview.net/forum?id=kklr_MTHMRQjG.

[18] S. Z. Li and A. K. Jain, Eds., *Handbook of Face Recognition*, 2nd ed. London, U.K.: Springer, 2011. DOI: 10.1007/978-0-85729-932-1.

[19] National Institute of Standards and Technology. "Face Recognition Technology Evaluation (FRTE) 1:1 verification", Accessed: Jun. 15, 2026. [Online]. Available: https://pages.nist.gov/frvt/html/frvt11.html.

[20] T. Fawcett, "An introduction to ROC analysis", *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006. DOI: 10.1016/j.patrec.2005.10.010.

[21] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: https://www.deeplearningbook.org/.

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning", *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. DOI: 10.1038/nature14539.

[23] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review", *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017. DOI: 10.1162/neco_a_00990.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks", in *Advances in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[25] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, 2014, pp. 1701–1708. DOI: 10.1109/CVPR.2014.220.

[26] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition", in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, 2019, pp. 4690–4699. DOI: 10.1109/CVPR.2019.00482.

[27] A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 4–20, 2004. DOI: 10.1109/TCSVT.2003.818349.

[28] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning", *Pattern Recognition*, vol. 84, pp. 317–331, 2018. DOI: 10.1016/j.patcog.2018.07.023.

[29] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. L. Yuille, "Mitigating adversarial effects through randomization", in *Proc. Int. Conf. Learn. Representations*, Vancouver, BC, Canada, 2018. [Online]. Available: https://openreview.net/forum?id=Sk9yuql0Z.

[30] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis, and D. H. Chau, "SHIELD: Fast, practical defense and vaccination for deep learning using JPEG compression", in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, London, U.K., 2018, pp. 196–204. DOI: 10.1145/3219819.3219910.