technische universität
dortmund

fakultät für informatik

Bachelor's Thesis

Comparative Analysis of Adversarial Attack Methods

Alain Yvan Ngeukeu Ngongang

September 4, 2025

Supervisors:

PD Dr. Frank Weichert

Prof. Dr. Heinrich Müller

Computer Science VII
Computer Graphics
TU Dortmund

Contents

Mathematical notation

Notation	Meaning
x_0	Clean RGB image, shape $3 \times H \times W$, pixels in $[0, 1]$.
$\|\cdot\|_\infty$	L_∞ norm (maximum absolute entry).
$\|\cdot\|_2$	L_2 norm (Euclidean distance).
$B_\infty(x_0, \epsilon)$	L_∞ ball around x_0: $\{x : \|x - x_0\|_\infty \leq \epsilon\}$.
$\Pi_S(\cdot)$	Projection onto a set S (e.g. $B_\infty(x_0, \epsilon)$).
$\text{clip}_{[0,1]}(\cdot)$	Clipping operator to valid pixel range $[0, 1]$.
$\text{sign}(\cdot)$	Element-wise sign function.

1 Chapter 1: Introduction

1.1 Motivation and Background

Modern image classifiers can be made to fail with tiny, carefully crafted pixel changes that are hardly visible to the eye. These *adversarial examples* raise a practical question: how robust is a model in realistic settings, and where is it most brittle?
Such vulnerabilities are not merely theoretical: they have been documented and discussed across multiple safety-critical domains.
In autonomous driving, for instance, small perturbations to a traffic sign have been shown to make a car misinterpret a stop sign as a speed-limit sign, or perturb lane markings so that the vehicle drifts out of its lane [Eyk+18].
In biometric authentication, adversarial patterns embedded in glasses or clothing can trick face recognition systems, while fingerprint or iris sensors may be bypassed with carefully manipulated input images [Sha+16].
In healthcare, imperceptible modifications to an X-ray or MRI scan can flip a model's diagnosis from malignant to benign, with potentially severe consequences for patient treatment [Fin+19].

In summary, adversarial attacks threaten not only emphsafety (e.g., autonomous vehicles, medical imaging), but also emphsecurity (biometrics).
What makes them especially dangerous is that the perturbations are nearly invisible to humans, yet catastrophic for machine learning models.

This thesis studies robustness under a standard, norm-bounded threat model in which an attacker perturbs an image $x_0 \in [0,1]^{3\times H\times W}$ within a small L_∞ budget ϵ, producing x with $\|x - x_0\|_\infty \leq \epsilon$. We compare three widely used attacks: *FGSM* (a single gradient step), *BIM* (iterative FGSM with clipping), and *PGD* (iterative with a random start and projection).

A second, equally important aspect is data imbalance. Real-world label distributions are often long-tailed: many classes are rare and only a few are frequent. A single overall robustness number can hide large differences across these strata. To surface those differences fairly, we group labels into frequency buckets (Rare/Medium/Frequent) and evaluate each bucket side by side.

We read results through two complementary lenses.

Model-impact metrics tell us how much the model degrades (e.g., Accuracy Drop, Relative Drop, Confidence Drop).

Attack/perceptual metrics describe the perturbations themselves (e.g., Attack Success Rate, size measured by a norm, and image-similarity measures such as SSIM/PSNR). Keeping these families separate makes the conclusions easier to interpret: how *hard* the model is hit versus how *big/visible* the change is.

Research questions. This thesis is guided by three central research questions:

1. **RQ1: Bucket Sensitivity.** Does adversarial robustness vary systematically across class-frequency strata (Rare, Medium, Frequent), or is it largely independent of how often a class appears in the training data? That is: Do models behave differently in terms of robustness when a class appears rarely versus frequently, or does the number of times a class is seen during training have no real effect on how vulnerable it is to adversarial attacks?

2. **RQ2: Attack Comparison.** At matched perturbation budgets, how do FGSM, BIM, and PGD differ in terms of effectiveness (accuracy degradation, confidence degradation, attack success rate) and stealth (perturbation size, perceptual similarity)?

3. **RQ3: Strength–Stealth Trade-off.** How does increasing attack strength shift the balance between model degradation and perceptual similarity, measured via ℓ_2, SSIM, and PSNR? .That is: If we let the attacker change the image more strongly, how does that affect the trade-off between fooling the model and still keeping the image looking natural?

Approach in brief. We form Rare/Medium/Frequent buckets from the dataset's empirical label counts (with a simple fallback to keep all buckets populated). We draw a fixed, stratified pilot subset (caps: Rare=100, Medium=50, Frequent=50) and reuse *the exact same images* for every attack and every ϵ. We then report bucketed results using the metric families above, plus a few qualitative examples for intuition. This design keeps compute tractable, results reproducible, and comparisons fair across both attacks and class strata.

1.2 Structure of the Thesis

Chapter 2 introduces the core concepts: adversarial examples, norm-bounded threat models, and the attack families studied here (FGSM, BIM, PGD). It also outlines the metric families (model-impact vs. attack/perceptual) and explains why long-tailed label distributions matter for robustness evaluation.
Chapter 3 details the experimental design used to instantiate the study: dataset snapshot and task, frequency-aware bucketing and the fixed pilot subset, model and preprocessing, threat model and attack schedules, the evaluation metrics, and the reporting protocol.
Chapter 4 presents the evaluation. We establish a clean baseline, compare attacks at a representative budget, study how robustness changes as ϵ increases, analyze differences across buckets, and visualize the strength–stealth trade-off. We also include qualitative examples, basic robustness checks, limitations, and a concise summary of findings.
Chapter 5 discusses implications, practical takeaways, threats to validity, and directions for future work, and concludes the thesis.
References list all sources cited. **List of figures** list all figures.

Having outlined the motivation, research questions, and overall approach, the next chapter introduces the theoretical foundations: neural networks, adversarial threat models, attack families, and evaluation metrics.

2 Foundations: Adversarial Robustness in Image Classification

Chapter overview

Building on the research questions introduced in Chapter 1, this chapter provides the necessary theoretical background. It explains the attack methods and robustness metrics that will later be applied in the experimental study (Chapter 3).

2.1 Neural Networks and Adversarial Examples

2.1.1 Neural Networks in Vision

Modern image classifiers typically take a three-channel RGB image as input and output a probability distribution over classes [LBH15]. Convolutional layers detect local patterns such as edges and textures, while deeper layers progressively build higher-level concepts [LBH15]. The final linear layer, followed by a softmax, converts extracted features into class probabilities [Guo+17].

During training, model weights are adjusted to minimize a classification loss on labeled images [LBH15]; during inference, the same preprocessing is applied and the network returns the top-1 predicted label together with its confidence score [Rus+15; Guo+17].

We focus on such discriminative classifiers because they are the de-facto deployment choice in computer vision systems [LBH15] and the primary targets of adversarial attacks [Sze+14; GSS15].

2.1.1.1 Discriminative vs. Generative Models

Discriminative models directly learn $p(y \mid x)$: given an image x, they predict its label y [NJ02]. Generative models instead learn $p(x)$ or the joint $p(x, y)$, enabling

them to model or synthesize data [NJ02]. For evaluating robustness to small input changes, discriminative classifiers provide the most direct and widely used signal—accuracy and confidence under perturbations—which is why this work focuses on them [Sze+14; GSS15].

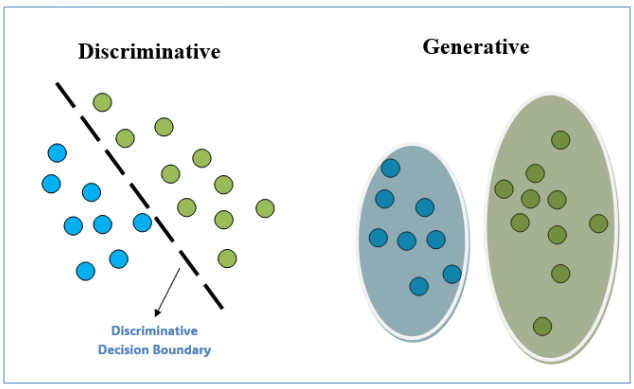

Figure 2.1: Discriminative models learn a decision boundary $p(y \mid x)$; generative models model the data or joint distribution $p(x)$ or $p(x, y)$.Source: TutorialsPoint, `https://www.tutorialspoint.com/gen-ai/discriminative-vs-generative-models.htm` (accessed 2025-08-23).

This focus on discriminative classifiers raises an important question: how vulnerable are such models to carefully designed perturbations of their input? The next subsection introduces the concept of adversarial examples and the threat models used to formalize this vulnerability.

2.1.2 Adversarial Examples: Concepts and Threat Models

Adversarial examples are inputs that look unchanged to humans but cause a trained classifier to make an incorrect prediction [Sze+14; GSS15]. They are created by adding a carefully chosen perturbation δ to a clean image x_0, producing $x = x_0 + \delta$ that fools the model while remaining visually similar to the original [Sze+14; GSS15]. To reason systematically about such adversarial manipulations, it is not enough to show that perturbations exist. We also need to specify the *rules of the game*: how much the input is allowed to change, what outcome the adversary is aiming for, and what information the adversary has about the classifier. These three ingredients — perturbation budget, attack goal, and attacker knowledge — define the threat model. The following subsections introduce each aspect in turn and motivate the specific setting adopted in this thesis.

2.1.2.1 Norm-Bounded Perturbations

To define what counts as a "small" perturbation, adversarial robustness research constrains the perturbation δ using a norm. A widely used constraint is the ℓ_∞ bound, which limits the maximum per-pixel change:

$$\|x - x_0\|_\infty \leq \epsilon,$$

where ϵ is a per-pixel budget on the normalized $[0, 1]$ scale. [Mad+18; CW17] Other ℓ_p norms (e.g., ℓ_2, ℓ_0) are also common. This formalization provides a clear, model-agnostic way to control the attack strength [Mad+18; CW17].
Defining such a perturbation budget specifies the space of admissible adversarial images. Within this space, however, different attack goals can be pursued: the adversary may simply want to cause any misclassification, or instead push the model toward a specific wrong label. We therefore next distinguish between untargeted and targeted attacks.

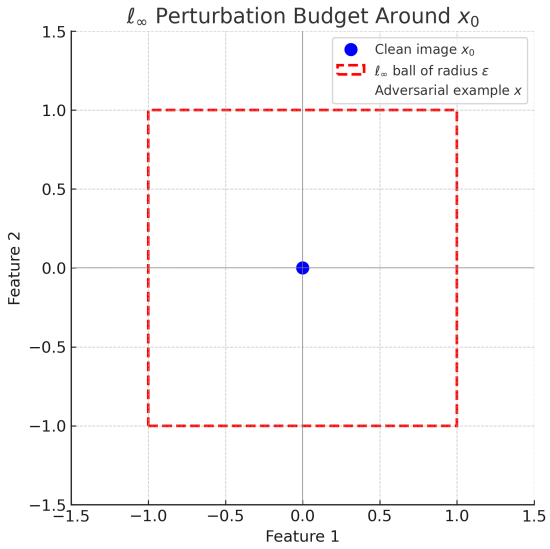

Figure 2.2: Illustration of an ℓ_∞ perturbation budget. The clean image x_0 is at the center, and the adversarial example x lies within the ℓ_∞ ball of radius ϵ (own illustration).

2.1.2.2 Untargeted vs. Targeted Attacks

Given the perturbation space defined above, different objectives can be specified for the adversary.

- **Untargeted attacks** aim to cause any misclassification (i.e., the model outputs an incorrect label different from y) [GSS15; KGB17a].

- **Targeted attacks** are more restrictive: they aim to force the model into predicting a specific incorrect label chosen in advance by the attacker [GSS15; KGB17a].

Untargeted attacks are typically easier to craft and directly capture the general vulnerability of a model. For this reason, this thesis employs the untargeted variants of Fast Gradient Sign Method (2.2.1), Basic Iterative Method (2.2.2), and Projected Gradient Descent (2.2.3), where the goal of the perturbation is to cause misclassification into any incorrect class rather than enforcing a specific target label.
Specifying the attack goal still leaves open the question of what the adversary actually *knows* about the model. This determines whether the perturbation can be computed directly from the model's gradients (white-box) or must be inferred indirectly (black-box), which we address next.

2.1.2.3 White-Box vs. Black-Box Settings

The adversary's knowledge of the model defines another important axis of the threat model.

- **White-box attacks** assume full access to the model's architecture, parameters, and gradients. This setting enables efficient gradient-based optimization and is considered the canonical *worst-case analysis* for robustness benchmarking [Mad+18].
- **Black-box attacks** assume no access to internal details. Here, the adversary must rely on queries to the model or on transfer attacks generated from a surrogate model [Pap+17].

Both are relevant in practice. However, this thesis focuses on gradient-based, white-box, norm-bounded attacks as the canonical setup for adversarial robustness evaluation.

Taken together, these three elements — a norm-bounded perturbation budget, untargeted attack goals, and white-box adversary knowledge — fully specify the threat model assumed throughout this work. With this foundation in place, we can now turn to the concrete attack families that instantiate these assumptions: the Fast Gradient Sign Method (FGSM), the Basic Iterative Method (BIM), and Projected Gradient Descent (PGD).

2.2 Attack Families

Building on the threat model defined in Section 2.1.2, we now turn to the concrete attack families studied in this thesis. While the threat model specifies *what* perturbations are admissible (e.g., bounded by an ℓ_∞ budget under white-box access), an attack method specifies *how* such perturbations are actually constructed.
At a high level, adversarial attacks take a clean image x_0 and construct a perturbed version $x = x_0 + \delta$ such that the classifier f misclassifies x, while the perturbation δ remains small and ideally imperceptible to humans. This principle was first highlighted by Szegedy et al. [Sze+14] and formalized by Goodfellow et al. [GSS15]. Modern benchmarks such as Madry et al. [Mad+18] adopt this formulation as the standard setup for evaluating adversarial robustness.

Formally, most gradient-based attacks can be expressed as an optimization problem:

$$\max_x \; L(f(x), y) \quad \text{subject to} \quad \|x - x_0\|_\infty \leq \epsilon,$$

where L is the cross-entropy loss, y is the true label, and ϵ is the perturbation budget controlling attack strength. The ℓ_∞ constraint ensures that each pixel is modified by at most ϵ, and clipping x to $[0, 1]$ guarantees valid image intensities.
Within this framework, three gradient-based attacks have become canonical benchmarks: the *Fast Gradient Sign Method (FGSM)*, the *Basic Iterative Method (BIM)*, and *Projected Gradient Descent (PGD)*. Together, they span a spectrum of attack strength: from a single-step perturbation to more refined iterative variants, culminating in PGD, which is widely regarded as the strongest first-order adversary.
For consistency, each attack is described below using a common structure:

- **Purpose and Motivation** — the intuition behind the attack and its intended role.
- **Formal Definition** — the update rule and mathematical formulation.
- **Interpretation and Intuition** — how the attack operates in practice.
- **Threat Model** — the assumed perturbation budget and attacker knowledge.

2.2.1 Fast Gradient Sign Method (FGSM)

2.1.5.1 Purpose and Motivation

Within the untargeted, white-box ℓ_∞ threat model introduced in Section 2.1.2, the Fast Gradient Sign Method (FGSM) provides a one–step procedure to craft an adversarial example by moving the input in the direction that most increases the loss, subject to the perturbation budget ϵ [GSS15]. Its appeal is computational efficiency: a single forward–backward pass suffices to expose a model's vulnerability [GSS15].

2.1.4.2 Formal Definition

Let $x_0 \in [0,1]^{3\times H\times W}$ be a clean image with true label y, and let f denote the classifier. We write $\tilde{x}_0 = (x_0 - \mu)/\sigma$ for channel–wise normalization using ImageNet means μ and standard deviations σ (cf. Section 3.4). FGSM constructs the adversarial example

$$x_{\text{adv}} = \text{clip}_{[0,1]}\Big(x_0 \;+\; \epsilon \cdot \text{sign}\Big(\nabla_x L(f(\tilde{x}_0),\, y)\Big)\Big), \tag{2.1}$$

where L is the cross-entropy loss, $\nabla_x L(\cdot)$ is the gradient of L with respect to the *input pixels* (backpropagated through the normalization), $\text{sign}(\cdot)$ applies the element-wise sign to the gradient, $\epsilon > 0$ is the ℓ_∞ perturbation budget, and $\text{clip}_{[0,1]}(\cdot)$ truncates each pixel to the valid range [GSS15].

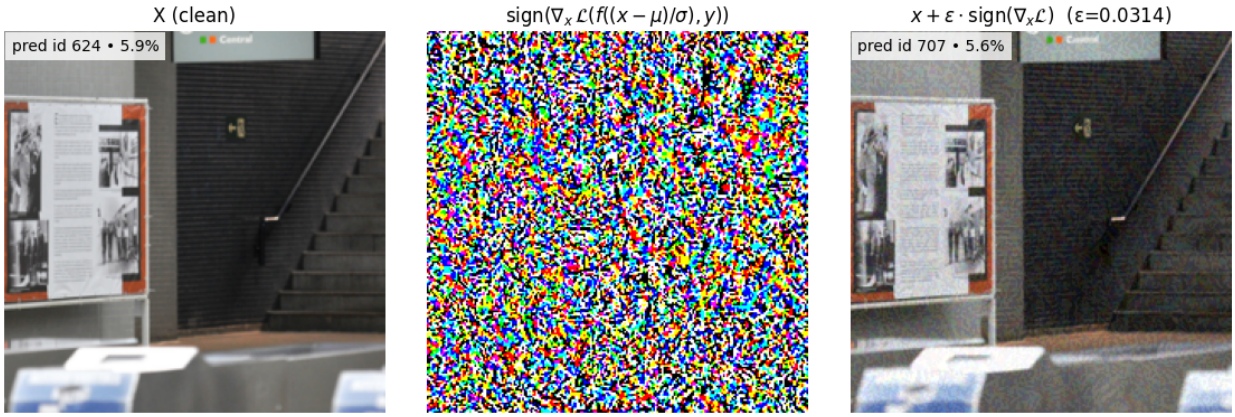

Figure 2.3: Fast Gradient Sign Method (FGSM) applied to a sample image. Left: the clean input x_0 with the model's top prediction. Middle: the signed gradient direction $\text{sign}(\nabla_x L)$, visualized for illustration. Right: the adversarial image $x_{\text{adv}} = x_0 + \epsilon \cdot \text{sign}(\nabla_x L)$ with $\epsilon = 0.0314$, which fools the classifier.(own illustration)

Equation (2.1) contains several components, each with a clear role in the attack:

- **Loss** $L(f(\tilde{x}_0), y)$**:** measures how badly the classifier f performs on the normalized input $\tilde{x}_0$ for the true label y. Increasing this loss pushes the model away from predicting y (misclassification pressure) [GSS15].
- **Gradient** $\nabla_x L$**:** the derivative of the loss with respect to the input pixels. It specifies, for each pixel, how the loss changes if that pixel is slightly increased or decreased. Following this gradient direction corresponds to the steepest increase in loss, which makes misclassification more likely [GSS15].
- **Sign operator** $\text{sign}(\cdot)$**:** applied element-wise to the gradient, producing entries in $\{-1, 0, 1\}$. This ensures each pixel is perturbed by exactly $\pm\epsilon$ in the most harmful direction permitted under the ℓ_∞ constraint. Goodfellow et al. prove that this update maximizes the linearized loss under an ℓ_∞ budget, which is why FGSM takes this form [GSS15].
- **Perturbation budget** ϵ**:** controls the maximum allowed pixel change under the ℓ_∞ threat model. Each pixel in the image can increase or decrease by at most ϵ (on the normalized $[0, 1]$ scale), so ϵ directly determines the attack strength: small values correspond to nearly imperceptible perturbations, while larger values produce more visible but also more effective adversarial examples [GSS15; Mad+18].
- **Clipping** $\text{clip}_{[0,1]}(\cdot)$**:** after adding the perturbation, pixel values might fall below 0 or above 1, which are invalid in normalized RGB images. Clipping projects every pixel back into the legal range $[0, 1]$, so that the adversarial example remains a valid image for both human viewing and model input [GSS15; KGB17a; Mad+18].

2.1.4.3 Interpretation and Intuition

FGSM is based on a simple idea: close to an image x_0, the model's loss surface can be approximated by a straight line [GSS15]. The gradient $\nabla_x L$ indicates in which direction the loss grows fastest. If each pixel may only change by a small amount ϵ (the ℓ_∞ budget), the strongest choice is therefore to push every pixel fully in that direction, using $\delta^\star = \epsilon \cdot \text{sign}(\nabla_x L)$.
As illustrated in Figure 2.3, the signed gradient highlights, for each pixel, whether increasing or decreasing its intensity will most strongly increase the loss. Adding ϵ

times this signed gradient to the clean image x_0 produces an adversarial example x_{adv} that can already fool the classifier after a single step.
This one-step update makes FGSM extremely fast, but less precise than iterative methods like BIM or PGD, which refine perturbations over multiple steps [Mad+18].

2.1.4.4 Threat Model

Throughout this work we apply the *untargeted* ℓ_∞ threat model from Section 2.1.2: each pixel may change by at most ϵ and the goal is any misclassification, not a specific wrong label [Mad+18; CW17]. Equation (2.1) implements exactly this setting with one gradient step computed at the normalized input $\tilde{x}_0$ and then mapped back to valid pixel space by clipping [GSS15].

FGSM thus serves as the simplest and fastest attack in our study. Its limitations — particularly the lack of refinement across multiple steps — motivated the development of iterative variants such as the Basic Iterative Method (BIM), which we discuss next , in line with the empirical patterns reported in Chapter 4 (cf. Sections 4.1.1 and 4.2.1).

2.2.2 Basic Iterative Method (BIM)

2.1.5.1 Purpose and Motivation

The Basic Iterative Method (BIM), also known as Iterative FGSM or I-FGSM, was introduced by Kurakin et al. [KGB17a] as a natural extension of the Fast Gradient Sign Method. While FGSM takes only a single large step in the gradient's direction [GSS15], BIM repeats the same operation with many smaller steps. This allows the perturbation to more carefully track the local loss surface and usually produces stronger adversarial examples.

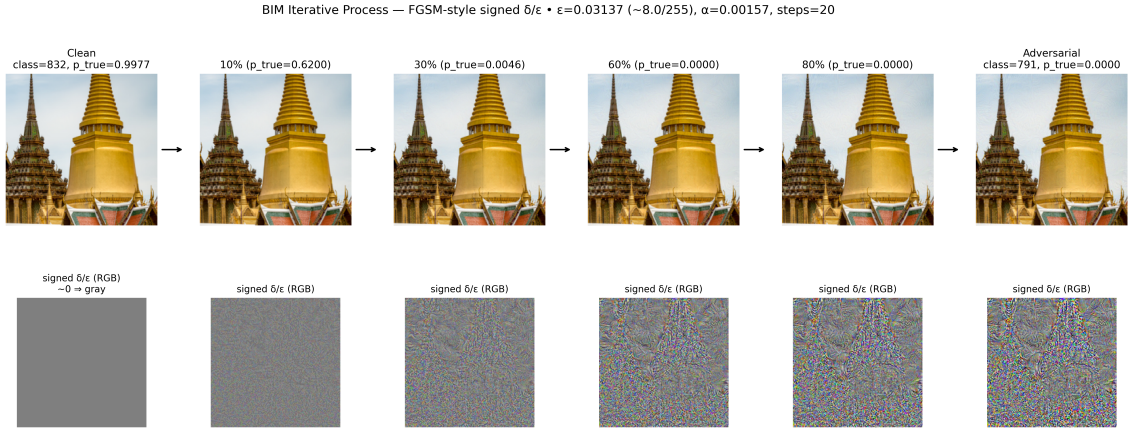

Figure 2.4: Basic Iterative Method (BIM) on a sample image with perturbation budget $\epsilon = 8/255$ and $T = 20$ iterations (step size chosen as $\alpha = \epsilon/T$ for a gradual build-up).
Top row: adversarial images at 0%, 10%, 30%, 60%, 80%, and 100% of the iterations; the model's true-label probability p_{true} decreases as small updates accumulate under the ℓ_∞ constraint.
Bottom row: visualizations δ/ϵ shown in RGB (each channel mapped from $[-1, 1]$ to $[0, 1]$). Early frames appear near gray (small $|\delta|$); later frames show increasingly saturated colors, indicating both the *direction* (sign) and *relative magnitude* of pixel changes, while remaining within the ℓ_∞ ball of radius ϵ around the clean image(own illustration).

2.1.5.2 Formal Definition

Starting from the clean input $x^{(0)} = x_0$, BIM performs the following iterative update:

$$x^{(t+1)} = \Pi_{B_\infty(x_0,\epsilon)\cap[0,1]}\Big(x^{(t)} + \alpha \cdot \operatorname{sign}\Big(\nabla_x L(f((x^{(t)} - \mu)/\sigma), y)\Big)\Big), \quad t = 0, \ldots, T-1.$$

Here:

- $\boldsymbol{x^{(t)}}$ is the *current image* after t update steps, with $x^{(0)} = x_0$ (the clean image).
- $\boldsymbol{f(\cdot)}$ is the *classifier*. We always feed it the *normalized* image $\tilde{x} = (x - \mu)/\sigma$, exactly as during training, so that gradients are meaningful for this model.
- $\boldsymbol{(\cdot - \mu)/\sigma}$ performs *per-channel normalization* (subtract mean μ and divide by std. σ for R/G/B).
- $\boldsymbol{L(f(\tilde{x}), y)}$ is the *loss* (cross-entropy) comparing the model's output on $\tilde{x}$ to the true label y.
- $\boldsymbol{\nabla_x L}$ is the *gradient of the loss with respect to the input pixels.* It tells, for each pixel, which small change would most increase the loss; moving in this

direction makes misclassification more likely [GSS15].

- $\mathbf{sign}(\cdot)$ takes the element-wise sign of the gradient, producing entries in $\{-1, 0, 1\}$.
- $\boldsymbol{\alpha}$ is the *step size*: how far one iteration moves along the signed gradient. Smaller α gives finer, safer moves; larger α moves faster but can overshoot.
- $\boldsymbol{\epsilon}$ is the *perturbation budget* for the ℓ_∞ threat model: *each pixel* may change by at most ϵ up or down (on the normalized $[0, 1]$ scale).
- $\boldsymbol{B_\infty(x_0, \epsilon)}$ is the ℓ_∞ *ball* (a hypercube) around x_0:

$$B_\infty(x_0, \epsilon) = \{\, x : \ \|x - x_0\|_\infty \leq \epsilon \,\}.$$

  Constraining $x^{(t)}$ to this set guarantees the per-pixel budget is respected *at every step*.

- $\mathbf{[0, 1]}$ is the *valid pixel range*. Because updates can push values slightly below 0 or above 1, we must clip them back so the image remains valid for both viewing and the model [KGB17a; Mad+18].
- $\boldsymbol{\Pi_{B_\infty(x_0,\epsilon)\cap[0,1]}(\cdot)}$ is the *projection* that enforces both constraints after each update: stay inside the ℓ_∞ ball *and* inside the pixel range. A practical element-wise form is

$$\Pi(z) = \mathrm{clip}_{[0,1]}\Big(x_0 + \mathrm{clip}(z - x_0,\, -\epsilon,\, +\epsilon)\Big),$$

  i.e., first cap the deviation from x_0 to $[-\epsilon, \epsilon]$ per pixel, then clip to $[0, 1]$ [Mad+18].

- $\boldsymbol{T}$ is the *number of iterations*. After T projected steps we output $x^{(T)}$ as the adversarial example. Larger T allows more refinement, at higher compute cost [KGB17a; Mad+18].

2.1.5.3 Interpretation and Intuition

BIM can be viewed as FGSM applied repeatedly with smaller steps. After each update, the perturbation is checked: if it leaves the ℓ_∞ ball or produces invalid pixel values, projection brings it back inside. This way, BIM behaves like a constrained optimization procedure, gradually climbing the loss surface until the image crosses the decision boundary [KGB17a].

As illustrated in Figure 2.4, the probability assigned to the true label gradually decreases as iterations proceed, while the perturbation δ accumulates in small, structured steps that remain within the ℓ_∞ budget. This visual progression highlights how BIM incrementally pushes the input across the decision boundary rather than relying on a single large step.
Because it takes multiple iterations, BIM is slower than FGSM but usually much more effective, since it can refine perturbations to exploit the model's vulnerabilities more thoroughly.

2.1.5.4 Threat Model

BIM is evaluated under the same untargeted ℓ_∞ threat model as FGSM. The constraint $\|x^{(t)} - x_0\|_\infty \leq \epsilon$ guarantees that the perturbation remains imperceptibly small at every step, while clipping to $[0, 1]$ maintains valid RGB values [Mad+18].

The attack stops after T iterations, returning $x^{(T)}$ as the adversarial example. Because it refines perturbations across multiple steps, BIM usually achieves stronger degradation than FGSM at the same budget — a pattern that will also become evident in Chapter 4 (cf. Sections 4.1.2 and 4.2.2). Its iterative nature, however, still lacks the randomization and stronger convergence guarantees of Projected Gradient Descent (PGD), which we discuss next.

2.2.3 Projected Gradient Descent (PGD)

2.1.6.1 Purpose and Motivation.

Projected Gradient Descent (PGD) was introduced by Madry et al. [Mad+18] as the most widely used benchmark attack for adversarial robustness. It extends the Basic Iterative Method (BIM) [KGB17a] by adding *random restarts* inside the perturbation set. While BIM always begins at the clean image x_0, PGD starts from a randomly perturbed point inside the ℓ_∞ ball of radius ϵ around x_0. This makes PGD harder to defend against, as it can escape local regions where BIM might otherwise get stuck.

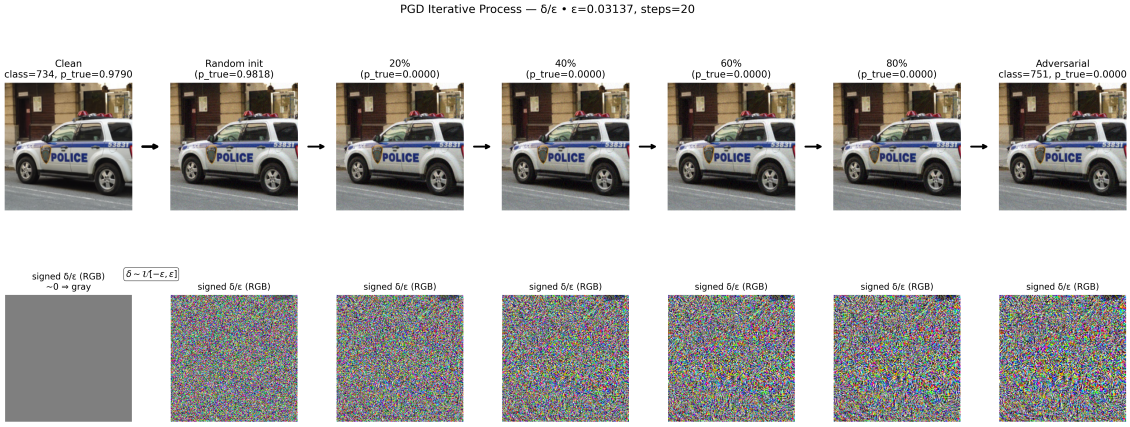

Figure 2.5: Projected Gradient Descent (PGD) applied to a high-confidence image with $\epsilon = 8/255$ and $T = 20$ iterations.
Top row: adversarial examples at selected milestones. Starting from a randomly initialized point ($\delta \sim \mathcal{U}[-\epsilon, \epsilon]$), each iteration applies a gradient step and projects the result back into the ℓ_∞ ball. The true-label probability p_{true} drops steadily while the predicted class eventually flips.
Bottom row: Perturbation maps $\text{sign}(\delta/\epsilon)$ in RGB, where gray 0 and saturated colors indicate large pixel-wise deviation in $[-\epsilon, \epsilon]$(own illustration).

2.1.6.2 Formal Definition.

Let x_0 be the clean input. PGD initializes with a random point

$$x^{(0)} = x_0 + \text{Uniform}(-\epsilon, \epsilon),$$

clipped to the valid pixel range $[0, 1]$. At each iteration, the update is

$$x^{(t+1)} = \Pi_{B_\infty(x_0,\epsilon)\cap[0,1]}\Big(x^{(t)} + \alpha \cdot \text{sign}\Big(\nabla_x L(f((x^{(t)} - \mu)/\sigma), y)\Big)\Big), \quad t = 0, \ldots, T-1.$$

Here:

- $\boldsymbol{x_0}$ is the *clean image* (RGB intensities normalized to $[0, 1]$). It is the center of the allowed perturbation region.
- $\boldsymbol{x^{(0)} = x_0 + \text{Uniform}(-\epsilon, \epsilon)}$ draws a *random starting point* for the attack by adding to each pixel an independent value sampled uniformly from $[-\epsilon, \epsilon]$, then clipping to $[0, 1]$. Because an ℓ_∞ ball is a hypercube, this sampling produces a valid random point inside the constraint set. The random start is the key difference to BIM and makes PGD more reliable: it avoids getting stuck in unfavourable regions by exploring different starting points inside the permitted set [Mad+18].

- $\boldsymbol{f(\cdot)}$ is the *classifier* that maps an input image to class scores (logits) or probabilities.

- $\boldsymbol{(\cdot - \mu)/\sigma}$ denotes *channel-wise normalization* (subtract mean μ and divide by standard deviation σ) applied before feeding the image to the model, exactly as during training (cf. Section 3.4). Gradients are computed through this normalization so that updates ultimately change the *original pixels* correctly.

- $\boldsymbol{L(f(\tilde{x}), y)}$ is the *loss function* (cross-entropy) comparing the model's output on the normalized input $\tilde{x} = (x - \mu)/\sigma$ to the true label y. A larger L means the model is doing worse on the intended class.

- $\boldsymbol{\nabla_x L(\cdot)}$ is the *gradient of the loss with respect to the input pixels*. It tells, for each pixel, in which direction a small change would most increase the loss. Moving along this direction makes misclassification more likely [GSS15].

- $\textbf{sign}(\cdot)$ applies the sign element-wise to the gradient, producing entries in $\{-1, 0, 1\}$. Under an ℓ_∞ constraint (per-pixel bound), the maximizer of the linearized loss is to push each pixel fully in the gradient's sign direction; this is why PGD (like FGSM/BIM) uses a sign step [GSS15].

- $\boldsymbol{\alpha}$ is the *step size* of each iteration. Intuitively, it is how far we move along the (signed) gradient in one step. Smaller α gives finer, more careful updates; larger α moves faster but risks overshooting. In practice, α is chosen relative to ϵ and the number of steps T (e.g., $\alpha \approx \epsilon/T$), so the total motion stays within budget [Mad+18].

- $\boldsymbol{\epsilon}$ is the *perturbation budget* for the ℓ_∞ threat model: *each pixel* may change by at most ϵ up or down. Thus ϵ is the "attack strength" knob: small ϵ is subtler/harder; larger ϵ is more visible/easier [GSS15; Mad+18].

- $\boldsymbol{B_\infty(x_0, \epsilon)}$ is the ℓ_∞ *ball* (a hypercube) around x_0:

$$B_\infty(x_0, \epsilon) = \{\, x : \ \|x - x_0\|_\infty \leq \epsilon \,\}.$$

  Constraining $x^{(t)}$ to this set enforces the per-pixel budget at *every* step.

- $\boldsymbol{[0, 1]}$ is the *valid pixel range*. Because adding noise can push values below 0 or above 1, we must bring them back to this legal range so the result is a valid image for the model and for visualization [KGB17a; Mad+18].

- $\boldsymbol{\Pi_{B_\infty(x_0,\epsilon)\cap[0,1]}(\cdot)}$ is the *projection operator* that enforces both constraints after each update step: stay inside the ℓ_∞ ball and inside the pixel range. Concretely, an equivalent element-wise implementation is

$$\Pi(z) \;=\; \mathrm{clip}_{[0,1]}\Big(x_0 \;+\; \mathrm{clip}(z - x_0,\, -\epsilon,\, +\epsilon)\Big),$$

  i.e., first limit the per-pixel deviation from x_0 to $[-\epsilon, \epsilon]$ and then clip the result to $[0, 1]$ [Mad+18].

- $\boldsymbol{T}$ is the *number of iterations*. After T projected steps we output $x^{(T)}$ as the adversarial example. Larger T lets the attack refine the perturbation more, at the cost of computation [Mad+18].

2.1.6.3 Interpretation and Intuition.

PGD can be understood as the culmination of the FGSM–BIM family of methods: like BIM it takes multiple small steps along the signed gradient, but unlike BIM it does not always begin at the clean image x_0. Instead, it starts from a random point within the allowed ℓ_∞ ball, so that each run explores a different region of the perturbation space. This randomization prevents the attack from being trapped in regions where the gradient is less effective, making PGD stronger and more reliable than BIM [Mad+18].
As illustrated in Figure 2.5, the random initialization ($\delta \sim \mathcal{U}[-\epsilon, \epsilon]$) places the starting point away from x_0, and subsequent iterations steadily decrease the true-label probability p_{true} until the predicted class flips. The bottom row shows how perturbations accumulate in structured, signed patterns, while always remaining within the ℓ_∞ budget.
Madry et al. describe PGD as the "universal first-order adversary," since it captures the strongest attack one can build from gradient information alone.

2.1.6.4 Threat Model.

PGD operates under the same untargeted ℓ_∞ white-box threat model as FGSM and BIM. The perturbation at every step is constrained by

$$\|x^{(t)} - x_0\|_\infty \leq \epsilon,$$

and the projection step ensures the result remains a valid image.

The random initialization makes the attack stochastic, so multiple restarts can be run to increase attack success. This added strength is confirmed in Chapter 4, where PGD consistently achieves the highest degradation across evaluation metrics.

Having introduced the main adversarial attacks considered in this work, we now turn to the question of how their impact can be systematically measured. Attacks like FGSM, BIM, and PGD generate perturbed images that degrade model performance, but the extent of this degradation must be quantified using appropriate metrics. To this end, the following section introduces a simple taxonomy of robustness metrics, grouped according to the aspect of robustness they capture.

2.3 Measuring Robustness: a Simple Metric Taxonomy

To judge robustness, we group metrics into three families that answer different questions:
(A) how the model's predictions change including Accuracy Drop , Relative Accuracy Drop and Confidence Drop ,
(B) how "successful" an attack is including Attack Success Rate (ASR), and
(C) how big or visible the perturbation is including ℓ_2 Perturbation Size , Structural Similarity Index (SSIM) ,and Peak Signal-to-Noise Ratio (PSNR) .

2.3.1 Accuracy Drop

Accuracy Drop is a core metric for measuring the degradation in a model's classification performance when it is exposed to adversarial perturbations. In the context of image classification, it quantifies how much worse a classifier performs on adversarial images compared to clean (unmodified) ones. The metric is widely used in adversarial robustness research and is derived from two standard quantities: clean accuracy and robust (adversarial) accuracy.

2.2.1.1 Clean Accuracy

The first ingredient of Accuracy Drop is the model's performance on unmodified images, referred to as *clean accuracy*. For each image, the classifier outputs a score for every possible class, which is converted into probabilities using the standard *softmax*

function [LBH15]. The class with the highest probability is called the *top-1 prediction*. If this top-1 prediction matches the true label, the classification is counted as correct. This definition of top-1 accuracy is the standard evaluation metric in large-scale image classification benchmarks, such as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Rus+15]. It therefore provides the natural baseline performance of the model before adversarial perturbations are applied.

2.2.1.2 Adversarial Accuracy (Robust Accuracy)

The second ingredient is the model's performance on adversarially perturbed images, often called *adversarial accuracy* or *robust accuracy*. The definition is identical to clean accuracy: we again consider the model's top-1 prediction and compare it to the true label. The difference is that here the evaluation is performed on inputs that have been deliberately modified by an adversarial attack. These adversarial images remain visually close to the originals but are constructed to push the model towards a wrong prediction.
This measure has become the standard way to report robustness in adversarial machine learning, starting from the PGD benchmark of Madry et al. [Mad+18] and later adopted in standardized robustness evaluations such as AutoAttack [CH20].

Taken together, clean accuracy and adversarial accuracy define the drop illustrated in Figure 2.6: in this example, a model with 90% clean accuracy achieves only 40% accuracy on adversarial inputs, leading to a 50 percentage point reduction. This gap is formally captured in the next subsection as *Accuracy Drop*.

2.2.1.3 Definition and Formula

Accuracy Drop combines the two previous quantities into a single metric of robustness harm. It measures how much worse a classifier performs on adversarially perturbed inputs compared to its performance on clean inputs.
Formally, for an evaluation set S and perturbation budget ϵ,

$$\text{AccuracyDrop}(S, \epsilon) \;=\; \text{Acc}_{\text{clean}}(S) \;-\; \text{Acc}_{\text{adv}}(S, \epsilon),$$

where:

- $\text{Acc}_{\text{clean}}(S)$ is the fraction of correctly classified images without perturbation (baseline performance).

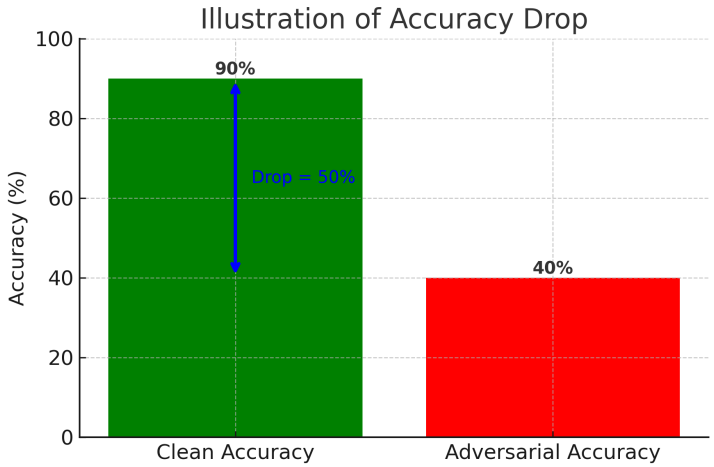

Figure 2.6: Illustration of Accuracy Drop: the clean accuracy (90%) drops to adversarial accuracy (40%) under attack, giving a 50 percentage point reduction.(own illustration)

- $\mathrm{Acc}_{\mathrm{adv}}(S, \epsilon)$ is the fraction of correctly classified images after adversarial perturbations of strength ϵ are applied.

2.2.1.3 Interpretation.

- Accuracy Drop quantifies the harm caused by adversarial perturbations.
- 0 means the attack had no effect (perfect robustness).
- Larger values mean the model is more vulnerable.

This difference between clean and adversarial accuracy has become the standard robustness metric in adversarial learning, starting with the PGD benchmark of Madry et al. [Mad+18] and adopted in standardized evaluations such as AutoAttack [CH20]. In this thesis, we report both adversarial accuracy itself ($\mathrm{Acc}_{\mathrm{adv}}$) and its difference to clean accuracy (Accuracy Drop), so that robustness can be assessed from both perspectives.

Since Accuracy Drop depends on the clean baseline, we next introduce Relative Accuracy Drop, which normalizes the loss by clean accuracy to enable fair comparison across models and buckets.

2.3.2 Relative Accuracy Drop

While Accuracy Drop gives the absolute decrease in performance, it does not account for differences in baseline (clean) accuracy across models. Relative Accuracy Drop normalizes the loss by clean accuracy, making the measure more comparable between classifiers with different starting points [Mad+18; CH20].
As illustrated in Figure 2.7, a clean accuracy of 90% combined with an adversarial accuracy of 40% produces the same absolute drop of 50 percentage points as before, but when normalized to the clean baseline this corresponds to a relative drop of about 55%. This shows how the metric highlights degradation in proportion to the model's original accuracy.

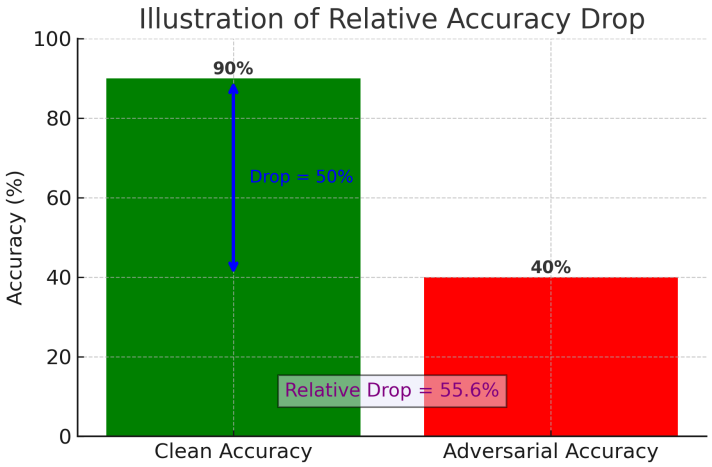

Figure 2.7: Illustration of Relative Accuracy Drop: with clean accuracy 90% and adversarial accuracy 40%, the drop is 50 points, corresponding to a relative drop of about 55%.(own illustration)

Formally, for evaluation set S and perturbation budget ϵ:

$$\mathrm{RelDrop}(S, \epsilon) = \frac{\mathrm{Acc}_{\mathrm{clean}}(S) - \mathrm{Acc}_{\mathrm{adv}}(S, \epsilon)}{\mathrm{Acc}_{\mathrm{clean}}(S)}.$$

Interpretation.

- RelDrop expresses the performance loss *relative* to the clean baseline.
- Example: if $\text{Acc}_{\text{clean}} = 90\%$ and $\text{Acc}_{\text{adv}} = 40\%$, then $\Delta\text{Acc} = 50$ points, but RelDrop $\approx 55\%$.
- This metric is useful when comparing models: a model with higher clean accuracy may suffer a larger absolute drop, but a smaller relative drop, showing better robustness.

Relative Accuracy Drop quantifies performance loss in percentage terms, but it still treats predictions in a binary way: correct or incorrect. What this view misses is how strongly the model believes in the correct label even when it remains correct. To capture this dimension of degraded certainty, we next turn to Confidence Drop, which tracks changes in the model's probability for the true class.

2.3.3 Confidence Drop

Confidence Drop is a robustness metric designed to measure how much a classifier's certainty in the correct class is reduced by adversarial perturbations [Mad+18]. The idea of using the model's softmax output as a proxy for confidence is supported by Hendrycks and Gimpel, who showed that the maximum softmax probability is a useful baseline for detecting misclassified or out-of-distribution inputs [HG17].
Unlike accuracy, which is a binary measure (correct or incorrect), *confidence* is a probabilistic score assigned by the model to each class, typically via the *softmax* function in the final layer of a neural network. This function transforms the raw output scores (logits) into probabilities that sum to 1.
The confidence assigned to the ground-truth class reflects how strongly the model believes the input belongs to that class. Therefore, when an adversarial perturbation is applied and this confidence drops—even if the prediction remains correct— it indicates that the model's internal representation has become less stable or more uncertain [Guo+17].
As illustrated in Figure 2.8, a clean input may initially have a true-label confidence of $p_{\text{true}} = 0.486$, which drops to $p_{\text{true}} = 0.001$ after perturbation with $\epsilon = 8/255$. This corresponds to a confidence drop of $\Delta p = 0.486 - 0.001 = 0.485$, demonstrating how adversarial noise reduces the model's certainty in the correct class. The following definition generalizes this idea to a whole evaluation set.

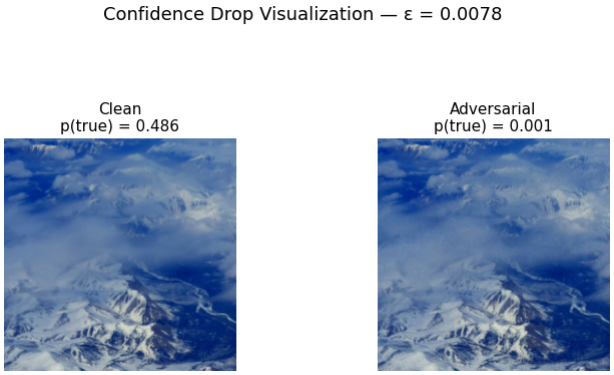

Figure 2.8: Illustration of Confidence Drop for True Class. The clean image (left) is initially predicted with a true-label confidence of $p(\text{true}) = 0.486$. After applying an adversarial perturbation with $\epsilon = 8/255$ (right), the model's confidence in the true class drops to $p(\text{true}) = 0.001$. This corresponds to a confidence drop of $\Delta p = 0.486 - 0.001 = \mathbf{0.485}$, illustrating how adversarial examples not only mislead the model but also reduce its certainty about the correct label.(own illustration)

2.2.3.1 Mathematical Definition

Let:

- $p^{\text{clean}}_{\text{true},i}$: the softmax confidence for the true class on clean input i
- $p^{\text{adv}}_{\text{true},i}$: the softmax confidence for the true class on adversarial input i
- S: a set of input samples (e.g., a class bucket or the whole evaluation set)

Then the Confidence Drop is defined as:

$$\text{ConfidenceDrop}(S) = \frac{1}{|S|} \sum_{i \in S} \left(p^{\text{clean}}_{\text{true},i} - p^{\text{adv}}_{\text{true},i} \right) \tag{2.2}$$

This metric gives a single value that represents, on average, how much less certain the model became—per image—about the correct label due to adversarial perturbations.
Explanation of terms:

- $\boldsymbol{p}^{\text{clean}}_{\text{true},\boldsymbol{i}}$ is the *confidence score* that the model assigns to the correct label (ground-truth class) when it is given the clean input i. This score comes from the *softmax probability* of the true class [Rus+15; Guo+17]. A higher value means the model is more certain that the input belongs to its true class.
- $\boldsymbol{p}^{\text{adv}}_{\text{true},\boldsymbol{i}}$ is the corresponding confidence score for the same true class when the input i has been adversarially perturbed. Because adversarial noise is designed to confuse the model, this confidence is usually smaller [Mad+18].

- $\boldsymbol{S}$ is the set of evaluation images over which the metric is computed (e.g., the full test set or a class-frequency bucket).
- $|\boldsymbol{S}|$ is the number of images in S. Dividing by $|S|$ means we are averaging across all inputs so that the metric represents the *mean drop in confidence per image*.
- The **difference** $\left(p^{\text{clean}}_{\text{true},i} - p^{\text{adv}}_{\text{true},i}\right)$ measures, for each image i, how much less certain the model became about the correct label after perturbation.
- The **average** $\frac{1}{|S|}\sum_{i\in S}$ aggregates this effect across the dataset, yielding one single value that represents the *average loss in certainty due to adversarial attacks* [HG17; CH20].

2.2.3.2 Interpretation

If a model is robust, its confidence in the true label should remain high even after perturbation. A large confidence drop suggests fragility: the model may still predict the correct label, but with much less certainty, indicating weaker calibration [Guo+17].

This makes Confidence Drop a valuable complement to accuracy, since it captures hidden uncertainty that accuracy alone cannot reveal. In Chapter 4, we use this metric to evaluate how FGSM, BIM, and PGD reduce the model's certainty across perturbation budgets and class-frequency buckets.

2.2.3.3 Softmax Confidence

The confidence values p_{true} used in this metric are obtained from the *softmax function*, which converts the model's raw output scores (logits) into probabilities that sum to one:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}.$$

The probability assigned to the true class is interpreted as the model's confidence, a convention widely used in evaluation and calibration studies [Guo+17; HG17].

While Confidence Drop highlights how adversarial perturbations erode a model's internal certainty, it does not directly capture how often the adversary actually succeeds in changing the prediction. To complement this defender-centered view, we next introduce the Attack Success Rate (ASR), an attacker-centered metric that measures the frequency of successful misclassifications.

2.3.4 Attack Success Rate (ASR)

The Attack Success Rate (ASR) is an attack-centric metric that measures how often an adversarial perturbation succeeds in changing the model's prediction [Sze+14; CW17; Pap+17; CH20]. Unlike accuracy-based metrics, which evaluate how much performance is retained under attack, ASR focuses only on the adversary's perspective: did the attack force the model into an error?

As illustrated in Figure 2.9, ASR is computed by comparing clean and adversarial predictions: if the adversarial input is misclassified, the attack counts as a success. In the example shown, 5 out of 6 perturbed images caused misclassification, corresponding to an ASR of 83.3% at $\epsilon = 8/255$.

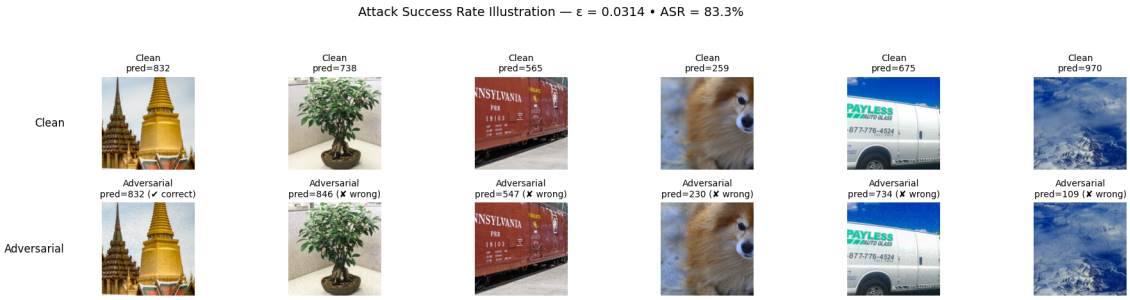

Figure 2.9: Attack Success Rate (ASR) Illustration. Each column shows a pair of images: the original clean input (top) and the adversarially perturbed version (bottom), with their respective predicted class IDs. An attack is considered successful if the adversarial image is misclassified (i.e., prediction differs from the true label). Checkmarks (correct) indicate robustness (prediction unchanged), while crosses (wrong) indicate successful attacks. In this example, **5 out of 6** adversarial examples caused misclassification, resulting in an ASR of **83.3%** at perturbation level $\epsilon = \frac{8}{255}$.(own illustration)

2.2.4.1 Mathematical Definition

For a set S of evaluation images with ground-truth labels y_i, and a classifier f, we define untargeted ASR at perturbation budget ϵ as

$$\mathrm{ASR}(S, \epsilon) \;=\; \frac{1}{|S|} \sum_{i \in S} \mathbb{1}\Big[f(x_i^{\mathrm{adv}}) \neq y_i \Big] ,$$

where x_i^{adv} is the adversarial example generated from clean input x_i, and $\mathbb{1}[\cdot]$ is an indicator that equals 1 if the adversary succeeded (prediction differs from the true label) and 0 otherwise. In our experiments, this value is multiplied by 100 to report ASR as a percentage (%) (cf. Section 3.6).

2.2.4.2 Interpretation

- ASR = 0 means no adversarial examples fooled the model (perfect robustness).
- ASR = 1 means every adversarial example caused misclassification (no robustness).
- Intermediate values quantify partial robustness: e.g., ASR = 0.82 means that 82% of adversarial inputs successfully fooled the model.
- Compared to accuracy-based metrics, which are *defender-centered* (how much performance is lost), ASR is *attacker-centered*: it measures how often the adversary achieves its goal.This complementary perspective highlights attack strength directly.

ASR has become a standard robustness benchmark [CH20]. In Chapter 4, we use it to compare FGSM, BIM, and PGD across perturbation budgets, showing how quickly each attack reaches high success rates.

While ASR captures how often adversarial attacks succeed, it does not reveal *how large* the underlying perturbations are. To complement this attacker-centered metric, we next examine the size and perceptual impact of perturbations, starting with the ℓ_2 norm.

2.3.5 ℓ_2 Perturbation Size (Pixel-Space Distance)

The metrics discussed so far — Accuracy Drop, Relative Accuracy Drop, Confidence Drop, and Attack Success Rate — all measure the *effect of adversarial examples on the model's behavior*. In contrast, perturbation-based metrics shift the focus to the *attack itself*: how large is the change made to the input in order to fool the classifier?

The ℓ_2 norm is the most common way to quantify this distortion. It was already used in the seminal work of Szegedy et al. [Sze+14], who introduced adversarial examples and measured their strength in terms of ℓ_2 distance from the clean image. Later, Carlini and Wagner [CW17] designed an attack that explicitly minimizes ℓ_2 distortion while still achieving high attack success.

As illustrated in Figure 2.10, increasing the perturbation budget ϵ produces adversarial images that look progressively more distorted, which is captured numerically by a larger ℓ_2 distance from the clean image.

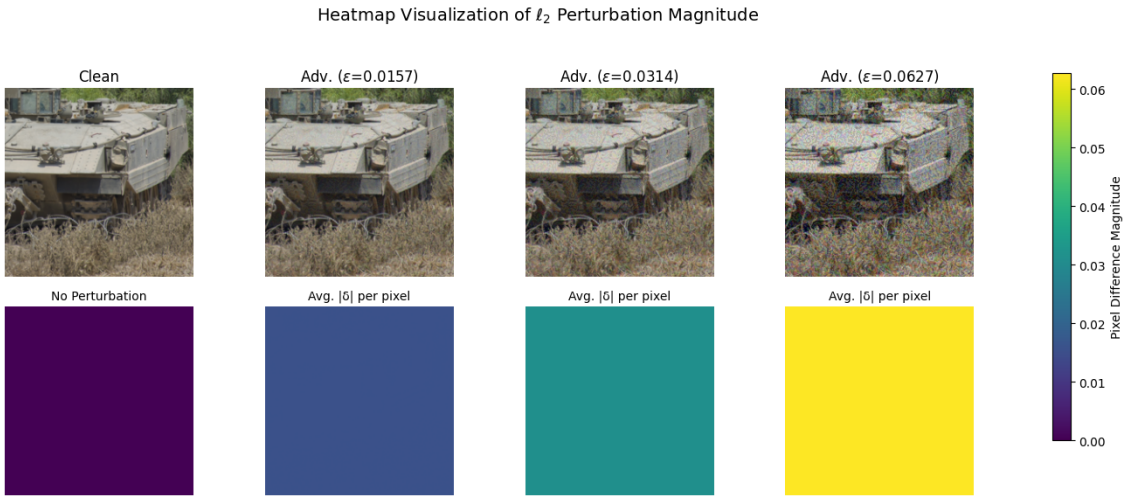

Figure 2.10: Visualization of adversarial perturbations using ℓ_2 norm.
Top row: Clean and adversarial images generated using FGSM with increasing ϵ (4/255, 8/255, 16/255).
Bottom row: Heatmaps showing per-pixel average absolute difference ($|\delta|$) from the clean image. As ϵ increases, the perturbation becomes more pronounced both visually and quantitatively.(own illustration)

2.2.5.1 Mathematical Definition

For a clean input image $x_0 \in \mathbb{R}^d$ and its adversarial counterpart x_{adv}, the ℓ_2 perturbation size is defined as the Euclidean distance between the two:

$$\|x_{\text{adv}} - x_0\|_2 \;=\; \sqrt{\sum_{j=1}^{d} \left(x_{\text{adv},j} - x_{0,j}\right)^2},$$

where d is the total number of pixels in the image.
Explanation of terms:

- $\boldsymbol{x_0}$: the clean image, represented as a d-dimensional vector of pixel intensities.
- $\boldsymbol{x}_{\text{adv}}$: the adversarial image obtained by adding a perturbation to x_0.
- $\boldsymbol{x}_{\text{adv},j} - \boldsymbol{x}_{0,j}$: the difference at pixel j, showing how much that pixel value was changed by the perturbation.
- $\sum_{j=1}^{d}(\cdot)^2$: squares and sums these per-pixel changes across all d pixels.
- $\sqrt{\cdot}$: takes the square root, giving the Euclidean distance between the two images in pixel space.

This metric thus produces a single scalar value: small ℓ_2 distances indicate subtle, imperceptible perturbations, while larger ℓ_2 values correspond to stronger distortions. The ℓ_2 norm was first used to quantify adversarial perturbations by Szegedy et al. [Sze+14] and later became the explicit optimization objective in the Carlini–Wagner attack [CW17].

2.2.5.2 Interpretation

The ℓ_2 norm measures the Euclidean distance between the clean and adversarial images, so it provides a direct sense of how much the input has changed.

- **Small ℓ_2 values:** correspond to perturbations that only slightly alter pixel values. For example, consider a grayscale image with four pixels: $x_0 = [0.50, 0.60, 0.55, 0.40]$ and its adversarial version $x_{\text{adv}} = [0.52, 0.59, 0.56, 0.42]$. The per-pixel changes are very small $(0.02, -0.01, 0.01, 0.02)$, and the ℓ_2 distance is

$$\|x_{\text{adv}} - x_0\|_2 = \sqrt{0.02^2 + (-0.01)^2 + 0.01^2 + 0.02^2} \approx 0.032.$$

Such a tiny ℓ_2 change would be imperceptible to humans, but could still mislead the classifier.

- **Large ℓ_2 values:** indicate stronger perturbations. If instead the adversarial image was $x_{\text{adv}} = [0.80, 0.30, 0.70, 0.10]$, the pixel changes are much larger $(0.30, -0.30, 0.15, -0.30)$, giving

$$\|x_{\text{adv}} - x_0\|_2 = \sqrt{0.30^2 + (-0.30)^2 + 0.15^2 + (-0.30)^2} \approx 0.56.$$

  Here the noise would be clearly visible, making the perturbation easier to detect but also making the attack more effective.

In Chapter 4, we report ℓ_2 perturbation size alongside accuracy-based metrics to compare how efficiently FGSM, BIM, and PGD achieve misclassification relative to the amount of distortion they introduce.

While the ℓ_2 norm quantifies pixel-space distortion, it does not always reflect human perceptual similarity. Two perturbations with the same ℓ_2 size can look very different to the eye. For this reason, robustness evaluation is often complemented by perceptual metrics such as SSIM and PSNR, which we discuss next.

2.3.6 Structural Similarity Index (SSIM)

Norm-based metrics such as ℓ_2 quantify the size of a perturbation in pixel space, but they do not always reflect how similar two images *appear* to the human eye. For instance, two images can have a relatively large ℓ_2 difference while still looking almost identical, or a small ℓ_2 distance but clearly visible structural distortions.

To address this limitation, Wang et al. [Wan+04] proposed the *Structural Similarity Index (SSIM)*, a perceptual quality metric that measures image similarity based on human visual perception. Unlike simple distance norms, SSIM compares images in terms of luminance, contrast, and structural correlation, which makes it more aligned with how humans judge visual similarity.

Subsequent studies such as Hore and Ziou [HZ10] have shown that SSIM is more reliable than traditional pixel-wise measures like PSNR, especially when evaluating subtle distortions such as adversarial perturbations.

2.2.6.1 Mathematical Definition

The Structural Similarity Index (SSIM) between two images x and y is computed locally over small patches and then averaged across the image. For a given patch, SSIM is defined as [Wan+04]:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)},$$

where:

- μ_x, μ_y: the mean pixel intensity of images x and y in the patch (represents *luminance*).
- σ_x^2, σ_y^2: the variance of pixel intensities (represents *contrast*).
- σ_{xy}: the covariance between x and y (represents *structural similarity* or correlation).
- C_1, C_2: small constants that stabilize the division, preventing numerical instability when denominators are close to zero.

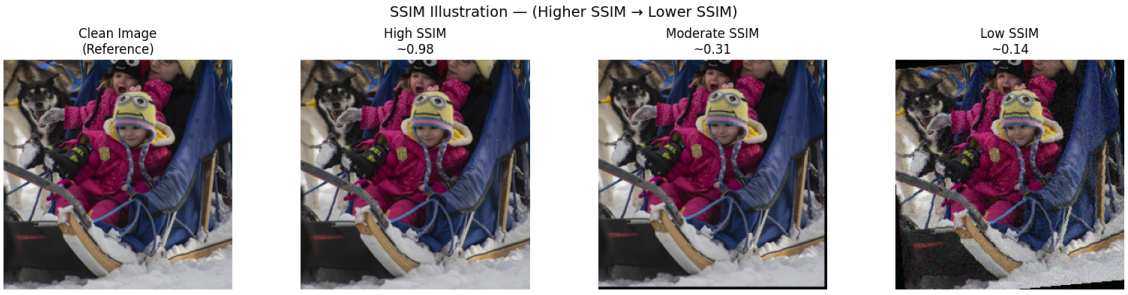

Figure 2.11: Illustration of SSIM values between a clean image and distorted versions. The clean reference image (leftmost) is compared to three degraded variants with decreasing structural similarity.
The second image has a high SSIM (~ 0.98), meaning it preserves most structures and edges.
The third image has moderate SSIM (~ 0.31), showing visible changes in contrast and local details.
The fourth image has low SSIM (~ 0.14), where structure and content are heavily distorted. SSIM values range from 1.0 (perfectly identical) to 0 (no structural similarity).(own illustration)

2.2.6.2 Interpretation

The Structural Similarity Index (SSIM) is designed so that its value corresponds to how similar two images appear to the human eye. The range is typically between 0

and 1, where 1 means the images are identical, and smaller values indicate increasing visual difference.
As shown in Figure 2.11, high SSIM values correspond to adversarial examples that are nearly indistinguishable from their clean counterparts, while lower values indicate progressively stronger visual degradation.

- **High SSIM (close to 1):** The adversarial example is nearly indistinguishable from the clean image. For example, if a clean image has a uniform background and an adversarial perturbation only changes pixel intensities by tiny amounts, the SSIM might be 0.98. Such a perturbation is imperceptible to humans, even though it may fool the classifier.
- **Moderate SSIM (around 0.6–0.8):** The images differ in noticeable ways but still share the same overall structure. For instance, adding stronger noise to an image may reduce SSIM to 0.65, at which point distortions become visible to the human observer.
- **Low SSIM (below 0.5):** The adversarial image is heavily distorted, making differences obvious. Such examples are less relevant in practice, since the perturbations are no longer subtle.

In Chapter 4, we report SSIM values to assess how visually similar adversarial images remain to their clean counterparts across FGSM, BIM, and PGD.

While SSIM captures structural similarity in a perceptually meaningful way, another widely used measure of visual distortion is the Peak Signal-to-Noise Ratio (PSNR), which expresses image quality in decibels. We discuss PSNR next.

2.3.7 Peak Signal-to-Noise Ratio (PSNR)

PSNR originates from signal processing, where it is used to measure the quality of a compressed or reconstructed image relative to a clean reference. [Wan+04; HZ10]
In the context of adversarial robustness, PSNR quantifies how much noise has been added to an image: higher PSNR values indicate that the adversarial example is closer to the clean image (less distortion), while lower PSNR values correspond to stronger perturbations. [Wan+04; HZ10]
Although PSNR is simpler and less perceptually aligned than SSIM, it remains a standard benchmark for comparing image quality and has often been reported alongside SSIM in studies of adversarial perturbations [Wan+04; HZ10].

2.2.7.1 Mathematical Definition

The Peak Signal-to-Noise Ratio (PSNR) is defined in terms of the mean squared error (MSE) between a clean reference image x_0 and an adversarial image x_{adv} [HZ10]:

$$\text{MSE}(x_0, x_{\text{adv}}) = \frac{1}{d}\sum_{j=1}^{d}(x_{0,j} - x_{\text{adv},j})^2,$$

$$\text{PSNR}(x_0, x_{\text{adv}}) = 10 \cdot \log_{10}\left(\frac{(\text{MAX}_I)^2}{\text{MSE}(x_0, x_{\text{adv}})}\right),$$

where:

- d is the number of pixels in the image.
- $x_{0,j}$ and $x_{\text{adv},j}$ are the intensities of pixel j in the clean and adversarial images.
- MAX_I is the maximum possible pixel value (e.g., 1.0 for normalized images or 255 for 8-bit images).
- MSE is the mean squared error, quantifying average distortion per pixel.
- The logarithm and scaling by 10 express the ratio in decibels (dB).

Numerical Example. Suppose two 8-bit grayscale images ($\text{MAX}_I = 255$) differ slightly, yielding an MSE = 4. The PSNR is then

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{255^2}{4}\right) \approx 42.1\,\text{dB}.$$

This is considered a high PSNR value, meaning the images are visually very similar. By contrast, if MSE = 50, then PSNR $\approx$ 31.1 dB, indicating stronger and more visible distortion.

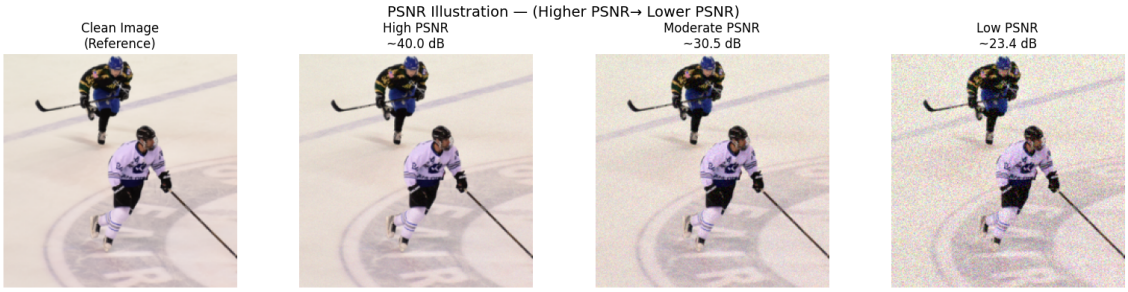

Figure 2.12: PSNR Illustration using additive noise. Visual comparison of a clean image (left) and its noisy versions with different noise levels. As noise increases, the images become more distorted and the PSNR values decrease. This illustrates how higher PSNR corresponds to cleaner images and lower PSNR to stronger perturbations.(own illustration)

2.2.7.2 Interpretation.

PSNR values are expressed in decibels (dB). Higher values correspond to smaller differences between the clean and adversarial image, while lower values indicate stronger distortions. As shown in Figure 2.12, increasing the level of additive noise decreases PSNR, making the perturbations more visible to the human eye.

- **High PSNR (> 40 dB):** The adversarial image is almost identical to the clean one. For example, with MSE $= 4$ on 8-bit images, PSNR ≈ 42 dB, which corresponds to perturbations that are imperceptible to the human eye.

- **Moderate PSNR (30–40 dB):** The adversarial perturbation is visible but not overwhelming. For instance, MSE $= 50$ yields PSNR ≈ 31 dB, where distortions can be noticed by humans but the image is still recognizable.

- **Low PSNR (< 30 dB):** The perturbation is strong and easily visible. At this level, adversarial examples no longer appear natural, which makes them less relevant for evaluating realistic robustness.

Unlike SSIM, which captures structural similarity in terms of luminance, contrast, and correlation, PSNR offers a straightforward scale in decibels that remains a standard for image quality reporting. In Chapter 4, we report PSNR values alongside SSIM to evaluate how visually similar adversarial examples remain to their clean counterparts across FGSM, BIM, and PGD.

2.3.8 Summary and Discussion of Metrics

The metrics in Section 2.3 provide complementary perspectives on adversarial robustness.

Accuracy-based. Accuracy Drop and Relative Accuracy Drop [Mad+18; CH20] measure how much performance is lost, while Attack Success Rate (ASR) [Sze+14; CW17; Pap+17] flips the view to the attacker's success. Together, they capture whether the model is right or wrong under attack.

Confidence-based. Confidence Drop [Guo+17; HG17] shows how much certainty in the correct label is eroded. This reveals hidden fragility even when predictions remain correct.

Perturbation- and perception-based. Norms such as ℓ_2 [Sze+14; CW17] measure the size of changes in pixel space, while SSIM and PSNR [Wan+04; HZ10] reflect how visible these changes are to humans. They indicate whether an attack remains realistic or produces obvious artifacts.

Overall. No single metric is sufficient. Accuracy- and confidence-based metrics describe model behavior, while norm- and perception-based metrics describe perturbation realism. Robustness analysis therefore requires reporting them side by side. This taxonomy underpins the empirical study in Chapter 3.

The concepts introduced here—adversarial threat models, attack families, and robustness metrics—form the foundation of the experimental design. In the next chapter 3, we describe how these concepts are operationalized into a concrete evaluation protocol.

3 Experimental Design

Chapter overview

Based on the adversarial methods and evaluation metrics introduced in Chapter 2, this chapter presents the methodology used to answer the research questions. It defines the dataset subset, bucketing procedure, model setup, and attack schedules that underpin the evaluation in Chapter 4.

3.1 Problem Setup and Goals

Adversarial examples—small, carefully chosen perturbations to input images—pose a practical risk to the reliability of deep learning systems. Even imperceptible modifications can cause state-of-the-art classifiers to output incorrect predictions with high confidence, as shown in early foundational work by Szegedy et al. and Goodfellow et al. [Sze+14; GSS15]. This undermines trust in neural networks, particularly in safety- or security-critical applications.

The objective of this thesis is to evaluate adversarial robustness in a controlled setting by comparing three representative gradient-based attack methods: FGSM, BIM, and PGD.

A second core dimension is class imbalance. Real-world datasets often exhibit long-tailed distributions, where a few classes dominate while many remain underrepresented [BMM18]. If robustness is only reported as a single overall accuracy under attack, systematic differences across these strata may remain hidden. This motivates a *frequency-aware protocol* that stratifies labels into *Rare*, *Medium*, and *Frequent* buckets.

The concrete goals of the experimental design are therefore threefold:

- **Bucket Sensitivity.** Does adversarial robustness vary systematically across class-frequency strata (Rare, Medium, Frequent), or is it largely independent of how often a class appears in the training data?(see for more details 5)
- **Attack comparison.** At matched budgets, how do FGSM, BIM, and PGD differ in terms of effectiveness (accuracy/confidence degradation, attack success rate) and stealth (perturbation size, perceptual similarity)? (see for more details 5)
- **Strength–stealth trade-offs.** How does increasing attack strength shift the balance between model degradation and perceptual similarity, measured via L_2, SSIM, and PSNR ? (see for more details 5)

This design ensures interpretability along two axes: **model impact** (accuracy, confidence, robustness drop) and **attack/perceptual characteristics** (success rate, perturbation visibility).

3.2 Dataset and Task

To instantiate the protocol, we use the **NIPS 2017 Adversarial Learning Development Set**, released as part of the NIPS 2017 competition on adversarial attacks and defenses (Google Brain) [KGB17b]. This dataset was explicitly designed for robustness studies, containing both natural images and ground-truth labels, and remains a standard benchmark for adversarial evaluation [Cro+21].
Each sample consists of:

- **ImageId** — a unique identifier corresponding to a PNG image file,
- **TrueLabel** — the correct class label, provided in the range $[1, 1000]$. For model compatibility, we convert labels to zero-based indexing $[0, 999]$.

The dataset spans 1000 diverse object categories, covering a wide range of natural and man-made entities. However, its distribution is highly imbalanced: many classes are represented by only a handful of samples, while others occur much more frequently. This long-tailed structure motivates the bucket-based evaluation introduced in Section 3.3.
Before proceeding with adversarial experiments, a set of integrity checks was performed:

- **Schema & completeness:** All expected fields are present with no missing values,
- **Identifier uniqueness:** No duplicate ImageId entries were found,
- **Label support:** Class labels cover the full 1–1000 range (converted to 0–999),
- **File consistency:** All referenced PNGs exist and spot-checks confirmed readability.

It is important to emphasize that the dataset is used only as a proof-of-concept instantiation of the broader experimental design. The methodology—frequency bucketing, stratified sampling, attack protocol, and evaluation metrics—is *dataset-agnostic* and transferable to other contexts.

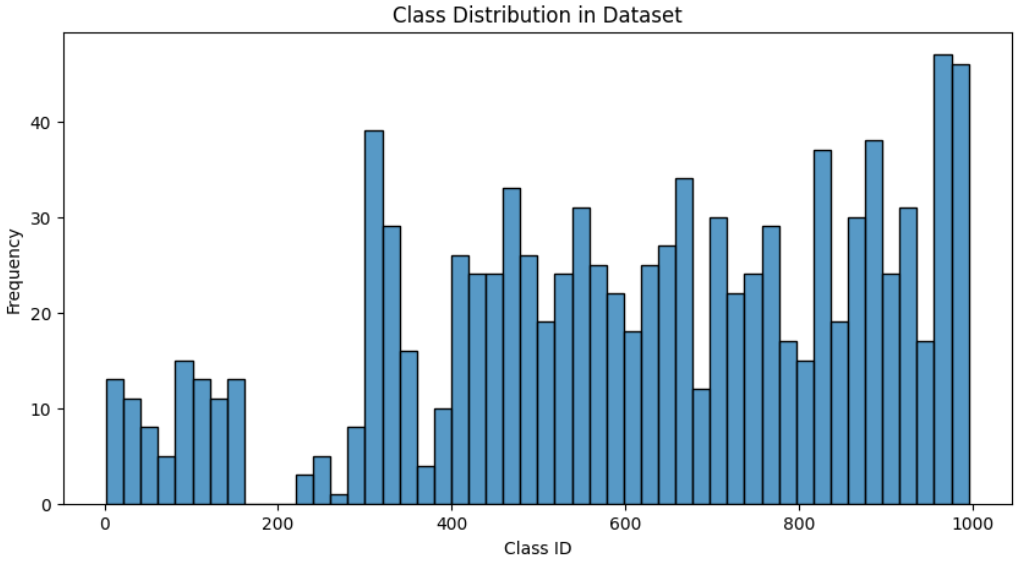

Figure 3.1: Histogram of class frequencies (class label vs. count), illustrating the long-tailed distribution of the dataset NIPS.(own illustration)

3.3 Frequency-Aware Bucketing and Pilot Subset

Real-world datasets often follow a long-tailed distribution: while some classes are well represented, many others appear only a few times. It is the same case for our dataset from NIPS 2017 Adversarial Learning Development Set(see Figure 3.1).Such imbalance can mask important differences in robustness if only aggregate performance is reported [BMM18]. To surface these differences, we adopt a frequency-aware protocol that groups classes into three buckets: *Rare*, *Medium*, and *Frequent*.

Formally, labels are sorted by their empirical sample counts and split into tertiles. As a safety fallback, if a bucket would remain empty, classes with at most one sample are assigned to *Rare*, those with two to three samples to *Medium*, and all others to *Frequent*. This guarantees that each bucket is populated. Applying this procedure to the NIPS 2017 snapshot yields the following distribution: (see Figure 3.2)

- **Rare:** 207 classes,
- **Medium:** 253 classes,
- **Frequent:** 540 classes.

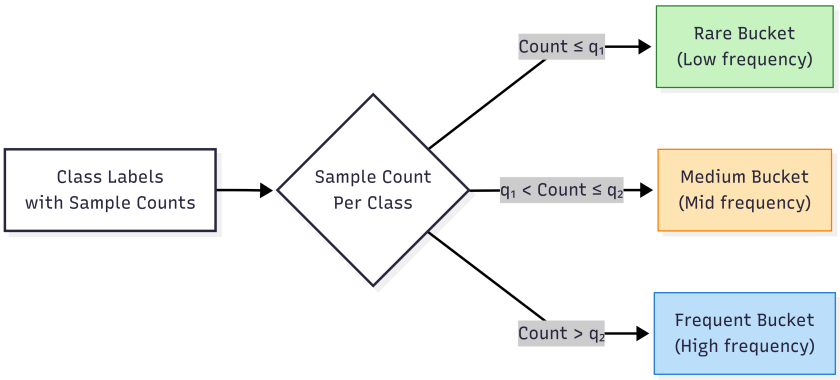

Figure 3.2: Schematic of the frequency-aware bucketing process: class labels are assigned to *Rare*, *Medium*, or *Frequent* buckets based on their empirical counts.(own illustration)

To make experiments tractable while preserving comparability, we construct a fixed pilot subset. Specifically, we draw 100 samples from the *Rare* bucket, 50 from *Medium*, and 50 from *Frequent*, for a total of 200 images. We draw the images once (no duplicates), and by fixing the random seed we ensure that the exact same subset can be reproduced in every run. The same pilot set is then reused across all attack methods and perturbation budgets, guaranteeing that comparisons are fair and controlled [CH20]. (see Figure 3.3)
This stratified design directly addresses the imbalance problem by enabling bucket-specific reporting while keeping the evaluation protocol lightweight and reproducible.

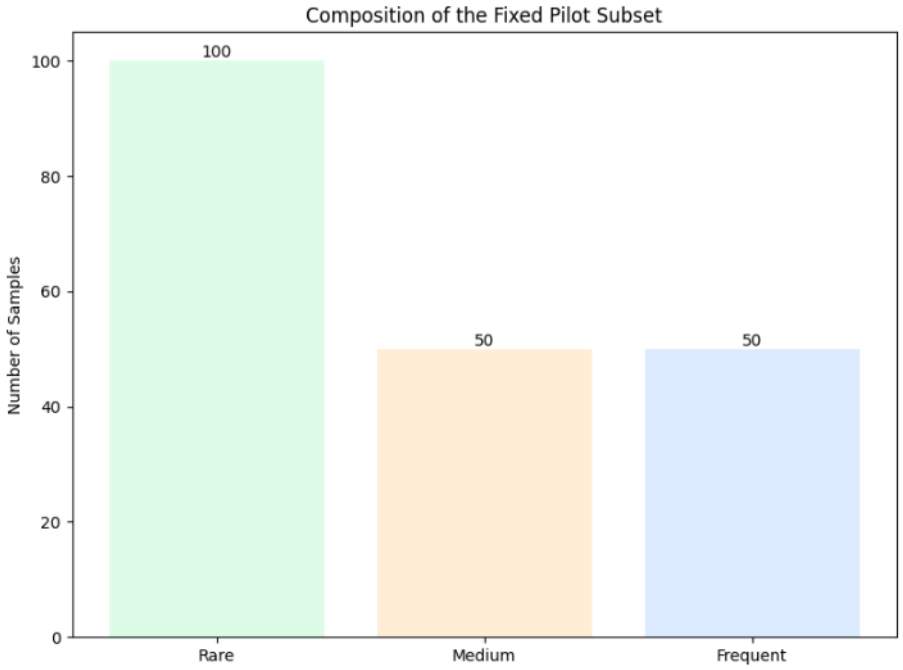

Figure 3.3: Composition of the fixed pilot subset: Rare = 100, Medium = 50, Frequent = 50 (total = 200 images).(own illustration)

3.4 Preprocessing and Model

All input images are RGB with values normalized to the $[0, 1]$ range. Before being passed to the classifier, each image is resized to 256×256 pixels and then center-cropped to 224×224. Pixel normalization is applied inside the model's forward pass using the standard ImageNet statistics: per-channel means $(\mu_R, \mu_G, \mu_B) = (0.485, 0.456, 0.406)$ and standard deviations $(\sigma_R, \sigma_G, \sigma_B) = (0.229, 0.224, 0.225)$. This ensures that inference is consistent with the original training conditions. (see Figure 3.4)

For all experiments, we use the **GoogLeNet** architecture [Sze+14], pre-trained on ImageNet. The model is held fixed and evaluated in `eval` mode, with no parameter updates or dropout applied. Inference runs on GPU if available. The preprocessing and model configuration remain identical across all attack methods and perturbation budgets. This design ensures consistency .

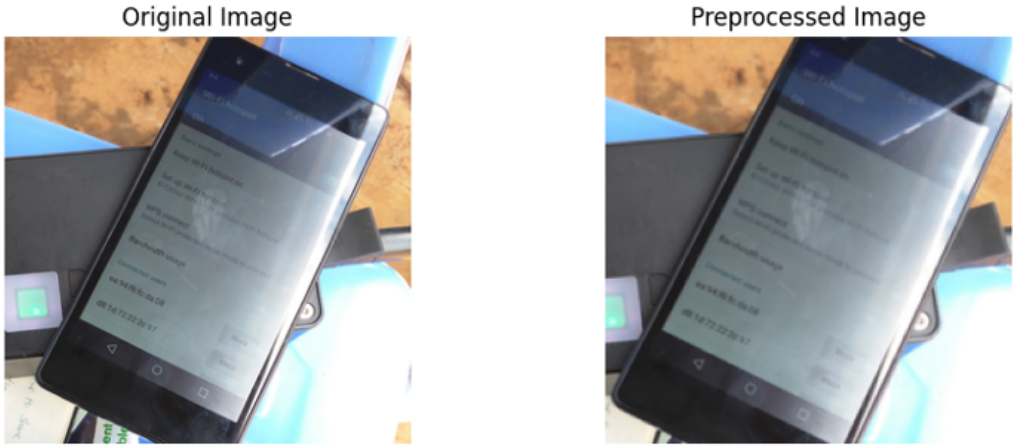

Figure 3.4: Example images before and after preprocessing. Left: original RGB input; right: normalized tensor after resize and crop.(own illustration)

3.5 Evaluation Setup and Attack Execution

To evaluate adversarial robustness in a controlled and reproducible manner, we apply all three attack methods—FGSM, BIM, and PGD—to the same stratified pilot subset introduced in Section 3.3. This subset consists of 200 images: 100 from the *Rare* bucket, 50 from *Medium*, and 50 from *Frequent*. Sampling is stratified and fixed, ensuring fair comparisons across methods and ϵ values.
Each attack is applied across a grid of seven perturbation budgets:

$$\epsilon \in \left\{0.00000, \tfrac{1}{255}, \tfrac{2}{255}, \tfrac{3}{255}, \tfrac{4}{255}, \tfrac{8}{255}, \tfrac{12}{255}\right\}$$

These values span a range from no perturbation (sanity check) to moderate-strength perturbations commonly used in the literature [KGB17a; Mad+18].
For every combination of image, attack, and ϵ, we generate an adversarial example and collect the following outputs:

- **Clean prediction:** class ID and softmax confidence for the true label,
- **Adversarial prediction:** class ID and its softmax confidence,
- **Perturbation statistics:** ℓ_2 norm, SSIM, and PSNR between clean and adversarial image,
- **Attack metadata:** attack name, ϵ value, and frequency bucket.

All perturbations are clipped to the $[0, 1]$ range to maintain valid pixel intensities, consistent with the L_∞ threat model defined in Section 2.1.2. The attacks operate

on pixel-space tensors that have been normalized using the procedure described in Section 3.4.
The attack methods follow the definitions from Chapter 2:

- **FGSM** (Section 2.2.1): a single-step gradient method,
- **BIM** (Section 2.2.2): an iterative extension of FGSM with projection,
- **PGD** (Section 2.2.3): an enhanced iterative method with random initialization.

All three methods operate under the same untargeted L_∞ threat model. For BIM and PGD, we use a fixed step size $\alpha = \epsilon/4$ and $T = 10$ iterations, unless otherwise specified.

3.5.1 Execution Protocol.

Each attack proceeds through the following standardized pipeline (see Figure 3.5):

1. Load pilot image x_0 and its true label y,
2. Normalize x_0 using ImageNet statistics,
3. Apply each attack at each ϵ, generating adversarial example x_{adv},
4. Run the classifier on both x_0 and x_{adv} to compute:
    - Accuracy Drop,
    - Relative Accuracy Drop,
    - Confidence Drop,
    - Attack Success Rate,
    - Perceptual metrics (ℓ_2, SSIM, PSNR).
5. Store all outputs for later analysis and visualization.

This unified evaluation protocol ensures consistency across attack types, perturbation levels, and class frequency strata. The outputs are stored in tabular format and analyzed in Chapter 4, where we examine degradation patterns, perceptual distortions, and tradeoffs between strength and stealth.

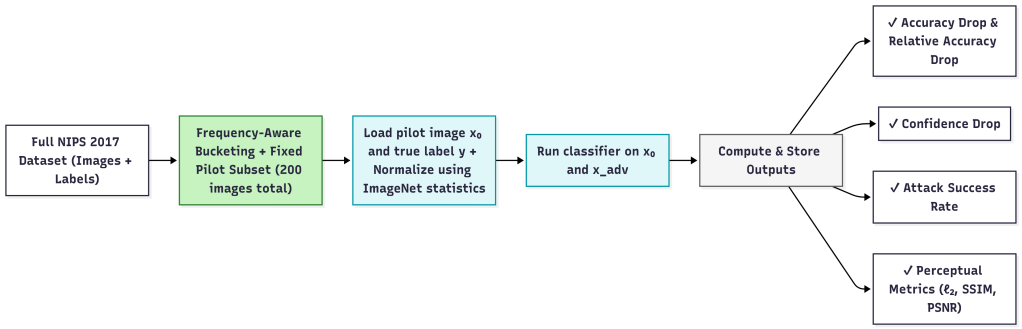

Figure 3.5: Execution protocol.(own illustration)

3.6 Evaluation Metrics

This section defines the key evaluation metrics used to assess adversarial robustness.

- **Correctness:** A binary indicator of whether the model's prediction matches the ground-truth label. This is computed both before and after the attack to determine the presence of misclassification (see Section 2.3.1).
- **Accuracy Drop:** The absolute difference in correctness between the clean and adversarial case. If the model was correct before but wrong after the attack, this value is 1; otherwise, 0 (Section 2.3.1).
- **Relative Accuracy Drop:** The drop in accuracy normalized by the original clean accuracy. This measures proportional degradation, especially informative when clean accuracy is less than 1 (Section 2.3.2).
- **Confidence Drop:** The difference in softmax probability assigned to the true label before and after the attack. A large drop suggests reduced model certainty under adversarial perturbation (Section 2.3.3).
- **Attack Success Rate (ASR):** The fraction of samples for which the model's adversarial prediction differs from the ground-truth label, regardless of whether the clean prediction was initially correct (Section 2.3.4).
- ℓ_2 **Norm:** The Euclidean norm of the perturbation vector, quantifying its magnitude in pixel space (Section 2.3.5).
- **SSIM (Structural Similarity Index):** A perceptual similarity score between the clean and adversarial image, sensitive to changes in structure, contrast, and luminance (Section 2.3.6).

- **PSNR (Peak Signal-to-Noise Ratio):** A global measure of distortion based on pixel-wise mean squared error; higher values indicate less perceptual change (Section 2.3.7).

All metrics are computed using the 0-based `TrueLabel0` field for consistency with the model's prediction index (see Section 3.2).

3.7 Summary and Experiment Blueprint

This section summarizes the experimental methodology from Chapter 3 and outlines the evaluation protocol used in Chapter 4. We evaluate three gradient-based attacks (FGSM, BIM, PGD) on a classifier trained on ImageNet labels, tested with a stratified pilot subset of 200 images from the NIPS 2017 dataset.
To account for class imbalance, labels were grouped into *Rare*, *Medium*, and *Frequent* buckets, ensuring fair, frequency-aware evaluation. Each attack was run across seven ℓ_∞ perturbation budgets, from imperceptible to visible changes, with both clean and adversarial predictions recorded. Metrics including accuracy, confidence drop, ℓ_2 norm, SSIM, and PSNR were computed per sample and aggregated by attack, perturbation strength, and frequency bucket.

With this protocol in place, we are now equipped to present and analyze the robustness results across attacks, budgets, and frequency strata in Chapter 4. This experimental blueprint ensures comparability across attacks and class-frequency buckets. The next chapter applies this design and reports empirical results for FGSM, BIM, and PGD across multiple robustness metrics.

4 Chapter 4: Evaluation

Chapter overview

Following the methodology defined in Chapter 3, this chapter presents the empirical evaluation.Each section corresponds to one of the robustness metrics introduced in Chapter 2, and together they provide the evidence needed to address RQ1–RQ3.

4.1 Accuracy Drop Across Attacks

This section reports the **absolute accuracy drop** $\Delta\text{Accuracy} = \text{Acc}_{\text{clean}} - \text{Acc}_{\text{adv}}$ across the three attacks—FGSM, BIM, and PGD—at varying ℓ_∞ perturbation budgets ϵ. The accuracy metric and its drop were defined in Section 2.3.1. Results are stratified by class frequency bucket (Rare, Medium, Frequent) to highlight differences in robustness depending on data imbalance (cf. Section 3.3).

4.1.1 Fast Gradient Sign Method (FGSM)

Figure 4.1 and Table 4.1 show the Accuracy Drop under FGSM. The effect is immediate and strong even at the smallest perturbation.

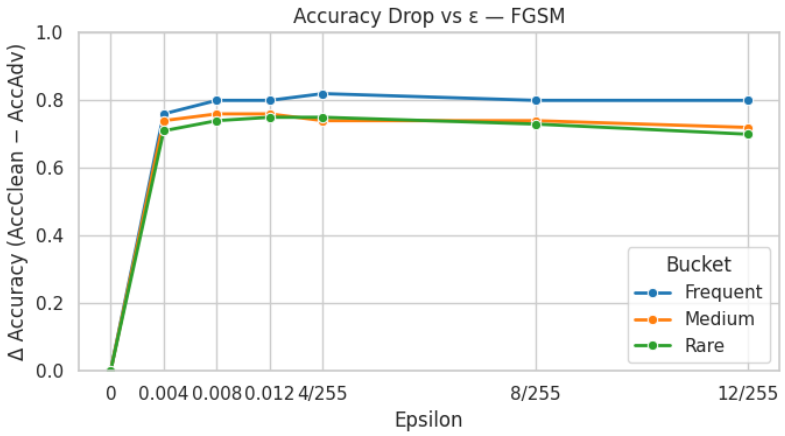

Figure 4.1: Accuracy drop under FGSM attack across buckets and perturbation strengths.

At $\epsilon = 1/255$, drops are already high: 0.76 for Frequent, 0.74 for Medium, and 0.71 for Rare. This means that more than 70% of the clean accuracy is lost with only the smallest possible pixel change. Increasing to $\epsilon = 2/255$ produces slightly higher drops of 0.80 (Frequent), 0.76 (Medium), and 0.74 (Rare). At $\epsilon = 3/255$ and above, the pattern stabilizes: Frequent and Rare stay around 0.80, while Medium remains the most affected bucket (0.12–0.16 drop). **Conclusion.** FGSM produces a large accuracy drop right away, but further increases in ϵ do not add much additional damage. This reflects FGSM's one-step nature: a single perturbation step is enough to disrupt many predictions, but it cannot refine beyond that.

Table 4.1: Accuracy Drop under FGSM by class-frequency bucket.

ϵ	Frequent	Medium	Rare
1/255	0.76	0.74	0.71
2/255	0.80	0.76	0.74
3/255	0.80	0.76	0.75
4/255	0.82	0.74	0.75
8/255	0.80	0.74	0.73
12/255	0.80	0.72	0.70

4.1.2 Basic Iterative Method (BIM)

Figure 4.2 and Table 4.2 show the Accuracy Drop under BIM. Unlike FGSM, BIM increases its effect gradually with ϵ, but reaches complete failure very quickly.

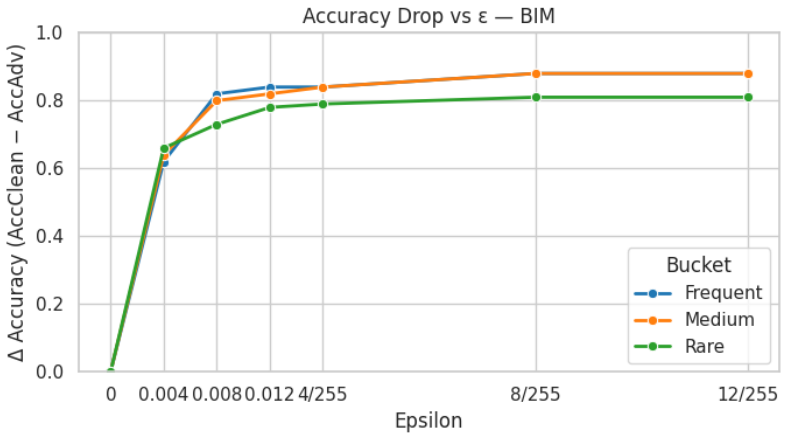

Figure 4.2: Accuracy drop under BIM attack across buckets and perturbation strengths.

At $\epsilon = 1/255$, drops are already substantial: 0.62 for Frequent, 0.64 for Medium, and 0.66 for Rare. At $\epsilon = 2/255$, the impact grows further: 0.82 (Frequent), 0.80 (Medium), and 0.73 (Rare). By $\epsilon = 3/255$, the drops reach 0.84 (Frequent), 0.82 (Medium), and 0.78 (Rare). At $\epsilon = 4/255$, all buckets are around 0.84, showing that most of the clean accuracy is already gone. At larger budgets the effect saturates: at $\epsilon = 8/255$ and $\epsilon = 12/255$, BIM drives the Frequent and Medium buckets to their maximum drop of 0.88, and the Rare bucket stabilizes at 0.81 (corresponding to its clean accuracy ceiling). **Conclusion.** BIM causes very large drops in accuracy already at small ϵ. Compared to FGSM, it reaches higher degradation at the same budget because its iterative nature allows it to refine perturbations. By $\epsilon = 4/255$, the model has already lost nearly all usable accuracy under BIM.

Table 4.2: Accuracy Drop under BIM by class-frequency bucket.

ϵ	Frequent	Medium	Rare
1/255	0.62	0.64	0.66
2/255	0.82	0.80	0.73
3/255	0.84	0.82	0.78
4/255	0.84	0.84	0.79
8/255	0.88	0.88	0.81
12/255	0.88	0.88	0.81

4.1.3 Projected Gradient Descent (PGD)

Figure 4.3 and Table 4.3 show the Accuracy Drop under PGD. PGD behaves very similarly to BIM, with rapid increases in drop values at small ϵ and full saturation at larger budgets.

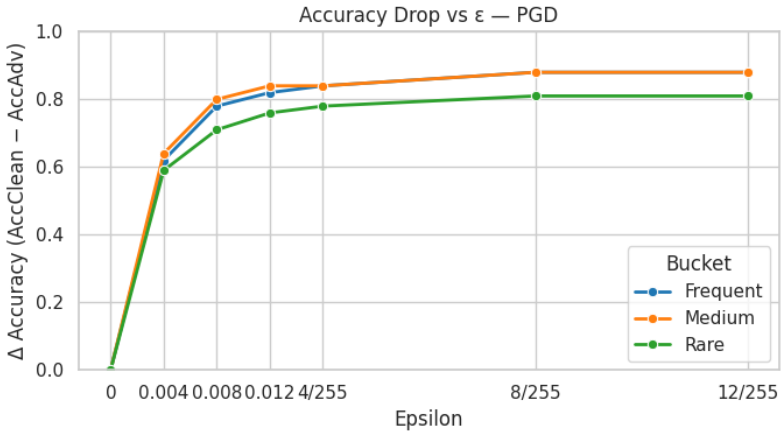

Figure 4.3: Accuracy drop under PGD attack across buckets and perturbation strengths.

At $\epsilon = 1/255$, drops are already large: 0.62 for Frequent, 0.64 for Medium, and 0.59 for Rare. At $\epsilon = 2/255$, the values increase to 0.78 (Frequent), 0.80 (Medium), and 0.71 (Rare). By $\epsilon = 3/255$, the drops reach 0.82 (Frequent), 0.84 (Medium), and 0.76 (Rare). At $\epsilon = 4/255$, they are almost identical across buckets: 0.84, 0.84, and 0.78.

For larger budgets, PGD saturates at the maximum: at $\epsilon = 8/255$ and $\epsilon = 12/255$, Accuracy Drop is 0.88 for Frequent and Medium (matching their clean accuracy of 0.88) and 0.81 for Rare (matching its clean accuracy of 0.81). **Conclusion.** PGD reduces accuracy very quickly and, like BIM, achieves complete degradation at modest ϵ values. Compared to FGSM, PGD is much stronger: while FGSM levels off around 0.80, PGD drives accuracy loss to the theoretical maximum. Differences between buckets are small, with Medium slightly more affected at low ϵ, but all converge to full failure as ϵ increases.

Table 4.3: Accuracy Drop under PGD by class-frequency bucket.

ϵ	Frequent	Medium	Rare
1/255	0.62	0.64	0.59
2/255	0.78	0.80	0.71
3/255	0.82	0.84	0.76
4/255	0.84	0.84	0.78
8/255	0.88	0.88	0.81
12/255	0.88	0.88	0.81

4.1.4 Comparison and Summary

Across all three attacks, Accuracy Drop increases very quickly with ϵ and then saturates at the clean accuracy ceiling of each bucket. FGSM produces a very steep decline already at $\epsilon = 1/255$ (drops of 0.71–0.76), but then levels off around 0.80. This plateau reflects FGSM's one-step nature: it causes immediate damage but cannot refine perturbations further. BIM and PGD, in contrast, both continue to drive accuracy down as ϵ grows. At $\epsilon = 1/255$, they already cause drops of 0.62–0.66. By $\epsilon = 2/255$, drops exceed 0.78 across buckets, and by $\epsilon = 4/255$ they reach ~ 0.84. At $\epsilon = 8/255$ and above, BIM and PGD achieve the maximum possible accuracy drop for each bucket (0.88 for Frequent/Medium, 0.81 for Rare), indicating total failure.

Overall, FGSM is less effective because it saturates early, while BIM and PGD are much stronger and eliminate almost all accuracy with only small perturbations. Differences between frequency buckets are small, typically within 0.05, though the Rare bucket sometimes drops slightly less due to its lower clean accuracy baseline.

While Accuracy Drop quantifies the absolute decrease in performance, it depends strongly on the clean baseline of the model or bucket. For example, losing 10 percentage points is far more severe for a class with only 20% clean accuracy than for one that starts at 90%. To make results comparable across different starting points, we next introduce Relative Accuracy Drop, which normalizes the loss by clean accuracy.

4.2 Relative Accuracy Drop

This section reports the *Relative Accuracy Drop* (RelDrop), defined in Section 2.3.2, using the same pilot subset and bucketing protocol as in Section 3.3. For each (attack, ϵ, bucket), RelDrop is computed as:

$$\text{RelDrop} = \frac{\text{Acc}_{\text{clean}} - \text{Acc}_{\text{adv}}}{\text{Acc}_{\text{clean}}},$$

where clean and adversarial accuracies are computed on the same images.

4.2.1 Fast Gradient Sign Method (FGSM)

Figure 4.4 and Table 4.4 show the Relative Accuracy Drop (RelDrop) under FGSM, stratified by frequency bucket. The values increase sharply already for very small perturbation budgets and then remain high across the entire range.

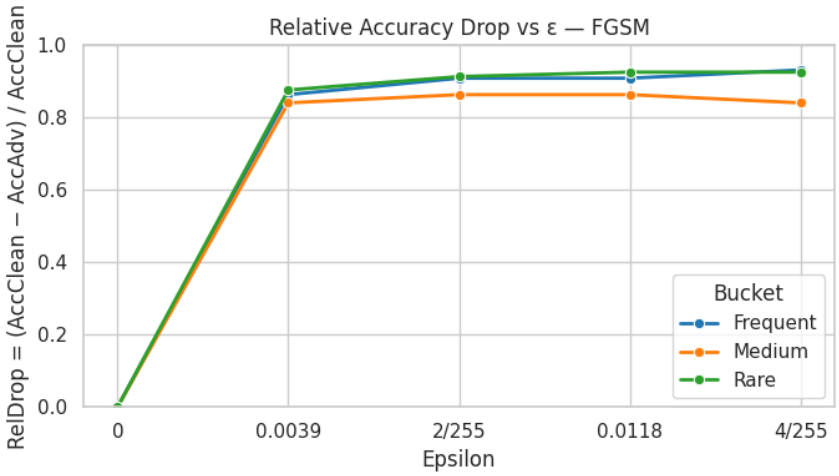

Figure 4.4: Relative Accuracy Drop under FGSM attack by frequency bucket.

At $\epsilon = 1/255$ (≈ 0.0039), RelDrop is already 0.86 for the frequent bucket, 0.84 for medium, and 0.88 for rare. This means that more than 85% of the model's clean accuracy is lost after pixel changes of less than 1/255 in intensity. By $\epsilon = 2/255$, RelDrop climbs further to 0.91 (Frequent), 0.86 (Medium), and 0.91 (Rare). At $\epsilon = 3/255$, values remain in the same range: Frequent 0.91, Medium 0.86, Rare 0.93. At $\epsilon = 4/255$, the Frequent bucket reaches its maximum (0.93), while Rare and Medium stay around 0.92 and 0.84. For larger budgets, saturation is visible: at $\epsilon = 8/255$ the RelDrop is still ≈ 0.91 for Frequent, 0.84 for Medium, and 0.90 for Rare. At $\epsilon = 12/255$, the values slightly decrease for Medium (0.82) and Rare (0.86), while Frequent remains at 0.91.

Conclusion. FGSM quickly drives Relative Accuracy Drop to about 0.9 within the smallest ϵ values, and further increases in ϵ only cause small fluctuations. The Rare bucket tends to degrade the most at small perturbations, while Frequent shows the highest RelDrop at $\epsilon = 4/255$. Overall, differences between buckets remain small (within ± 0.05), showing that FGSM produces rapid and broadly similar degradation across all frequency groups.

Table 4.4: Relative Accuracy Drop (RelDrop) under FGSM by class-frequency bucket.

ϵ	Frequent	Medium	Rare
1/255	0.8636	0.8409	0.8765
2/255	0.9091	0.8636	0.9136
3/255	0.9091	0.8636	0.9259
4/255	0.9318	0.8409	0.9259
8/255	0.9091	0.8409	0.9012
12/255	0.9091	0.8182	0.8642

4.2.2 Basic Iterative Method (BIM)

Figure 4.5 and Table 4.5 show the Relative Accuracy Drop (RelDrop) under BIM. Compared to FGSM, BIM causes even faster and more consistent degradation across all class-frequency buckets.

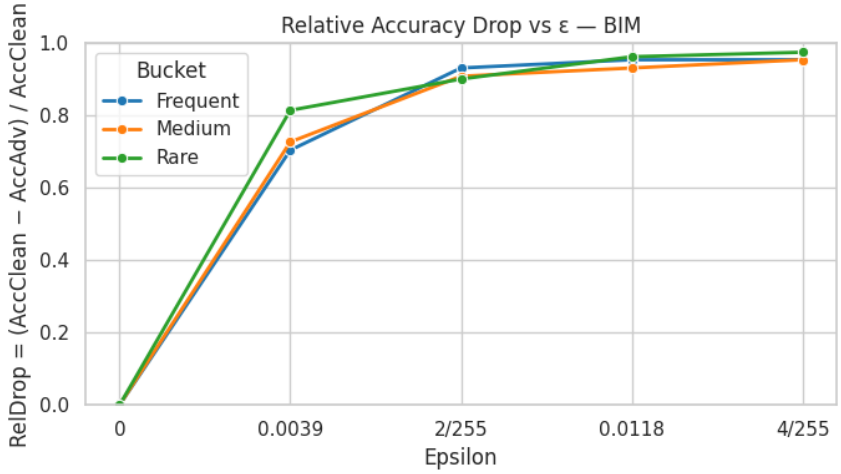

Figure 4.5: Relative Accuracy Drop under BIM attack by frequency bucket.

At $\epsilon = 1/255$, the Relative Drop is already very high: 0.70 for Frequent, 0.73 for Medium, and 0.81 for Rare. This means that between 70% and 80% of clean accuracy is lost almost immediately. By $\epsilon = 2/255$, values rise above 0.90 across all

buckets (0.93 Frequent, 0.91 Medium, 0.90 Rare). At $\epsilon = 3/255$, RelDrop increases further to 0.95 for Frequent, 0.93 for Medium, and 0.96 for Rare. At $\epsilon = 4/255$, Medium catches up with Frequent at 0.95, while Rare reaches its maximum of 0.98.

For larger budgets, saturation occurs: at $\epsilon = 8/255$ and $\epsilon = 12/255$, all three buckets have RelDrop = 1.00, meaning that accuracy is completely destroyed.

Conclusion. BIM very quickly drives Relative Accuracy Drop close to 1.0. Already at $\epsilon = 2/255$, more than 90% of clean performance is lost in every bucket, and by $\epsilon = 4/255$, almost total failure is reached. The Rare bucket is often the most affected (0.96–0.98), but the differences between buckets are relatively small. Overall, BIM is clearly stronger than FGSM in pushing all class groups to near-complete degradation.

Table 4.5: Relative Accuracy Drop (RelDrop) under BIM by class-frequency bucket.

ϵ	Frequent	Medium	Rare
1/255	0.7045	0.7273	0.8148
2/255	0.9318	0.9091	0.9012
3/255	0.9545	0.9318	0.9630
4/255	0.9545	0.9545	0.9753
8/255	1.0000	1.0000	1.0000
12/255	1.0000	1.0000	1.0000

4.2.3 Projected Gradient Descent (PGD)

Figure 4.6 and Table 4.6 show the Relative Accuracy Drop (RelDrop) under PGD. PGD produces very rapid and consistent degradation, similar to BIM but with slightly different dynamics across frequency buckets.

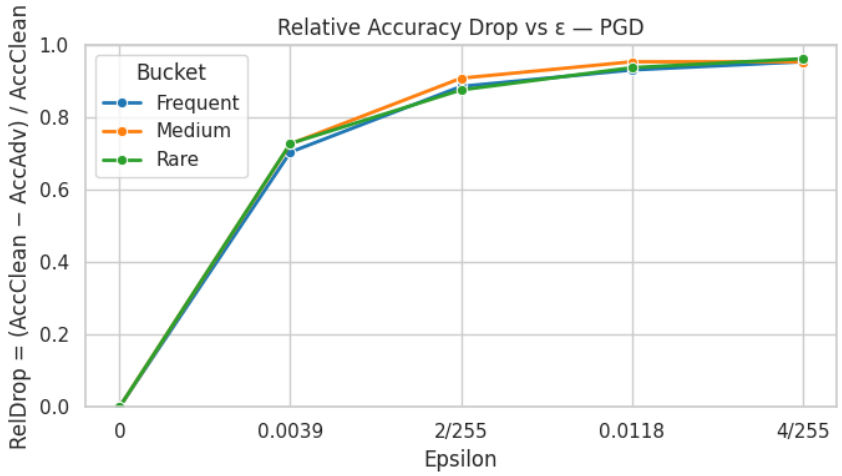

Figure 4.6: Relative Accuracy Drop under PGD attack by frequency bucket.

At $\epsilon = 1/255$, Relative Drops are already high: 0.70 for Frequent, 0.73 for Medium, and 0.73 for Rare. This means that around 70% of clean accuracy is lost with the smallest perturbation step. By $\epsilon = 2/255$, the values increase sharply to 0.89 (Frequent), 0.91 (Medium), and 0.88 (Rare), showing that PGD achieves over 85–90% degradation almost immediately.

At $\epsilon = 3/255$, the model loses virtually all of its accuracy: RelDrop is 0.93 for Frequent, 0.95 for Medium, and 0.94 for Rare. At $\epsilon = 4/255$, all three buckets are at 0.95 or above, with Rare again the most affected at 0.96.

For larger budgets, saturation occurs: at $\epsilon = 8/255$ and $\epsilon = 12/255$, all frequency buckets reach RelDrop = 1.00, indicating complete loss of accuracy.

Conclusion. PGD, like BIM, quickly drives Relative Accuracy Drop close to 1.0. Already at $\epsilon = 2/255$, the model has lost nearly all of its usable accuracy, and by $\epsilon = 4/255$, degradation is essentially complete. Across buckets, Medium and Rare classes are sometimes slightly more affected, but the overall differences are very small. PGD is thus one of the most effective attacks in producing total model failure with minimal perturbation budgets.

Table 4.6: Relative Accuracy Drop (RelDrop) under PGD by class-frequency bucket.

ϵ	Frequent	Medium	Rare
1/255	0.7045	0.7273	0.7284
2/255	0.8864	0.9091	0.8765
3/255	0.9318	0.9545	0.9383
4/255	0.9545	0.9545	0.9630
8/255	1.0000	1.0000	1.0000
12/255	1.0000	1.0000	1.0000

4.2.4 Comparison and Summary

Across all three attacks, Relative Accuracy Drop increases extremely quickly and then saturates. For FGSM, RelDrop rises above 0.85 already at $\epsilon = 1/255$ and stabilizes around 0.90 thereafter, with only small differences between frequency buckets. BIM and PGD are even stronger: at $\epsilon = 1/255$, both already cause around 70–80% loss of clean accuracy, and by $\epsilon = 2/255$ they exceed 0.90 in every bucket. At $\epsilon = 3/255$ to $4/255$, values are typically above 0.95, and from $\epsilon = 8/255$ onwards all three buckets reach 1.00, indicating complete accuracy loss.

Overall, FGSM produces rapid but slightly less complete degradation, while BIM and PGD drive the model to near-total failure at very small perturbation budgets. Differences between frequency buckets (Rare, Medium, Frequent) are modest across all methods, usually within ± 0.05, but Rare and Medium sometimes show slightly higher drops. These results confirm that relative degradation saturates very quickly: most of the clean accuracy is destroyed in the lowest part of the ϵ range, and additional perturbation budget adds little extra harm.

While Relative Accuracy Drop reveals how quickly overall performance collapses under attack, it still treats predictions in a binary way—either correct or incorrect. To capture this more nuanced dimension of robustness, we next turn to Confidence Drop, which measures how much the model's certainty in the true label is reduced by adversarial perturbations.

4.3 Confidence Drop at Fixed Perturbation

Figure 4.7 and and Table 4.7 shows the mean Confidence Drop across frequency buckets. Confidence Drop measures how much the model's softmax probability for the correct class decreases when inputs are perturbed (cf. Section 2.3.3).

The results show that all three attacks significantly reduce model confidence, even when predictions remain correct. On average, FGSM reduces confidence by 0.62 in the Frequent bucket, 0.59 in the Medium bucket, and 0.55 in the Rare bucket. BIM has an even stronger effect, with drops of 0.66 (Frequent), 0.63 (Medium), and 0.58 (Rare). PGD closely matches BIM, also producing drops of 0.66, 0.63, and 0.58 respectively.

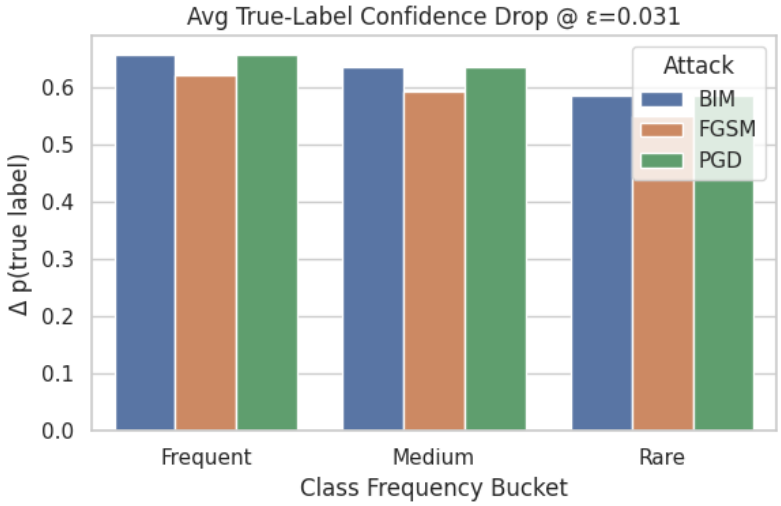

Figure 4.7: Average true-label confidence drop Δp_{true} at $\epsilon = 0.031$ ($\approx 8/255$), grouped by attack and frequency bucket.

Conclusion. Confidence is consistently eroded across all attacks and frequency buckets. Iterative methods (BIM, PGD) reduce confidence slightly more than FGSM, but the differences are small (typically less than 0.05). Across buckets, Frequent and Medium classes tend to lose slightly more confidence than Rare, but the overall pattern is that adversarial perturbations make the model much less certain in its predictions, regardless of class frequency.

Table 4.7: Mean Confidence Drop across attacks and class-frequency buckets.

Attack	Frequent	Medium	Rare
FGSM	0.619	0.593	0.549
BIM	0.655	0.633	0.584
PGD	0.655	0.633	0.584

While Confidence Drop reveals how adversarial perturbations undermine the model's certainty, it does not indicate whether this loss of confidence ultimately translates into a wrong prediction. To capture the frequency of actual misclassifications caused by adversarial examples, we next consider the Attack Success Rate (ASR).

4.4 Attack Success Rate (ASR)

Attack Success Rate (ASR) measures how often the model's prediction is changed by an adversarial perturbation (cf. Section 2.3.4). At $\epsilon = 0$, the adversarial input is identical to the clean input, so ASR should in principle be zero. However, in practice our model already misclassifies about 15.5% of clean inputs. This baseline error appears as a non-zero ASR at $\epsilon = 0$, and simply reflects the fact that ImageNet classifiers are not perfect: typical top-1 accuracies for models such as GoogLeNet or ResNet are only 70–85%, leaving error rates of 15–30% on clean images [Rus+15; HG17].

Figure 4.8 and Table 4.8 summarize the ASR trends across perturbation budgets. Under FGSM, the ASR rises steeply to 92% at $\epsilon = 0.0157$ (4/255), but then plateaus slightly lower around 88–90% for larger ϵ values. This shows that FGSM can reach high success quickly, but is less stable at larger perturbation sizes.

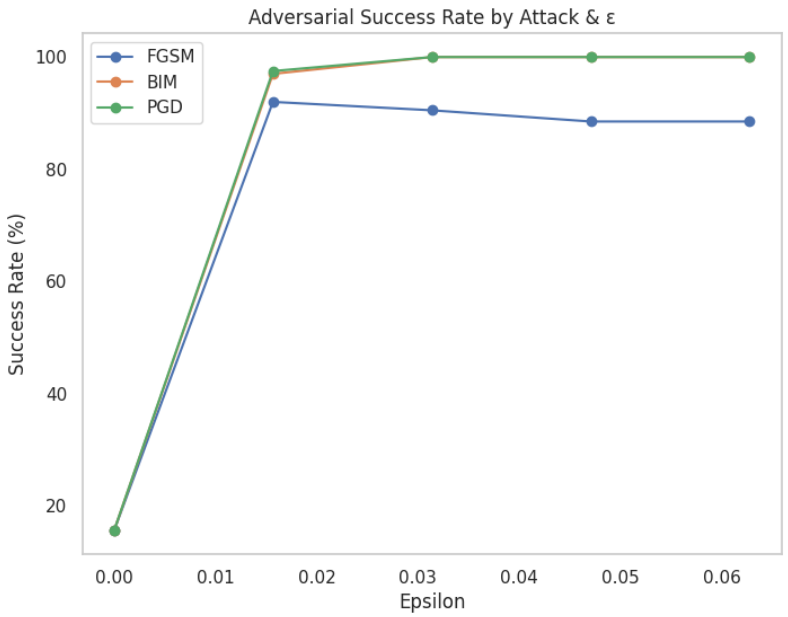

Figure 4.8: Attack success rate (ASR) by perturbation budget ϵ.

Iterative methods perform much more strongly. BIM achieves 97% ASR already at $\epsilon = 0.0157$ and reaches 100% from $\epsilon = 0.0314$ onwards, maintaining full success at all larger budgets. PGD is nearly identical: 97.5% at $\epsilon = 0.0157$ and 100% from $\epsilon = 0.0314$ upwards.

Conclusion. FGSM is effective but less reliable at higher ϵ, while BIM and PGD achieve complete success almost immediately and remain at 100%. The non-zero baseline at $\epsilon = 0$ is not due to the attack, but corresponds to the model's natural clean error rate.

Table 4.8: Attack Success Rate (ASR) across perturbation budgets. The non-zero baseline reflects the model's clean error rate.

Attack	ϵ	ASR (%)
FGSM	0	15.5
	0.0157	92.0
	0.0314	90.5
	0.0471	88.5
	0.0627	88.5
BIM	0	15.5
	0.0157	97.0
	0.0314	100.0
	0.0471	100.0
	0.0627	100.0
PGD	0	15.5
	0.0157	97.5
	0.0314	100.0
	0.0471	100.0
	0.0627	100.0

While Attack Success Rate captures how frequently adversarial perturbations succeed, it does not reveal how much the input has to be changed in order to achieve that success. Two attacks may both reach nearly 100% ASR, yet one might require much larger and more visible distortions than the other. To quantify this cost of success, we next examine perturbation size using the ℓ_2 norm.

4.5 Perturbation Size (ℓ_2 norm)

To quantify the magnitude of pixel changes introduced by adversarial attacks, we compute the average ℓ_2 norm of the perturbation (cf. Section 2.3.5). This value grows with ϵ, but the growth pattern differs strongly between attacks.

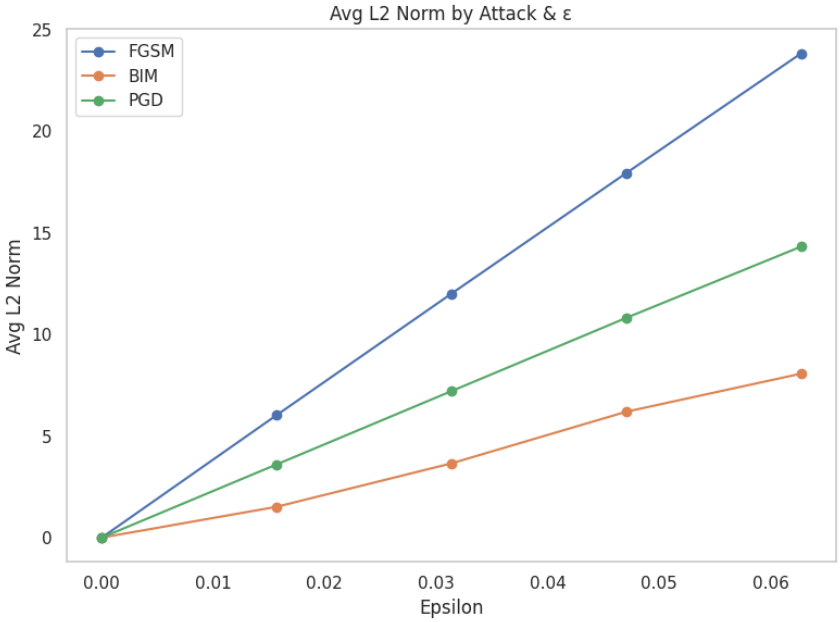

Figure 4.9: Average ℓ_2 perturbation norm across attacks and perturbation levels ϵ.

Table 4.9 and Figure 4.9 summarize the results. For FGSM, perturbation sizes increase rapidly with ϵ: at $\epsilon = 0.0157$, the average ℓ_2 norm is already 6.0, and by $\epsilon = 0.0627$ it reaches 23.8. This reflects FGSM's one-step nature: it modifies all pixels simultaneously, leading to large total distortions.

BIM, in contrast, achieves success with much smaller perturbations. At $\epsilon = 0.0157$, its ℓ_2 norm is only 1.5, less than one-quarter of FGSM's size at the same budget. Even at $\epsilon = 0.0627$, BIM remains at 8.1, about one-third of FGSM's distortion.

PGD lies between these extremes. At $\epsilon = 0.0157$, its perturbation size is 3.6 (larger than BIM, smaller than FGSM), and at $\epsilon = 0.0627$ it grows to 14.3. This reflects PGD's balance between FGSM's "all at once" update and BIM's fine-grained iterative steps.

Conclusion. FGSM perturbs images the most aggressively, BIM achieves high attack success with the smallest changes, and PGD strikes a middle ground. Thus, ℓ_2 analysis reveals not just whether an attack succeeds, but how *efficiently* it succeeds in terms of distortion magnitude.

Table 4.9: Average ℓ_2 perturbation size across attacks and budgets.

Attack	ϵ	Avg. ℓ_2 norm
FGSM	0.0157	6.03
	0.0314	12.01
	0.0471	17.95
	0.0627	23.84
BIM	0.0157	1.52
	0.0314	3.66
	0.0471	6.20
	0.0627	8.07
PGD	0.0157	3.60
	0.0314	7.21
	0.0471	10.82
	0.0627	14.33

While ℓ_2 perturbation size captures how large the changes are in pixel space, it does not necessarily reflect how those changes appear to a human observer. Two perturbations with the same ℓ_2 norm can look very different in practice— one may be imperceptible, while another introduces clear visual artifacts. To account for this perceptual dimension, we next turn to SSIM and PSNR, which measure similarity in terms of human visual perception.

4.6 Perceptual Similarity (SSIM, PSNR)

To assess how visible adversarial perturbations are to humans, we evaluate the Structural Similarity Index (SSIM) and Peak Signal-to-Noise Ratio (PSNR) (cf. Sections 2.3.6–2.3.7). High SSIM (≈ 1.0) and high PSNR (measured in dB) indicate that the adversarial image is perceptually very close to the clean one, while lower values indicate stronger visible distortion.

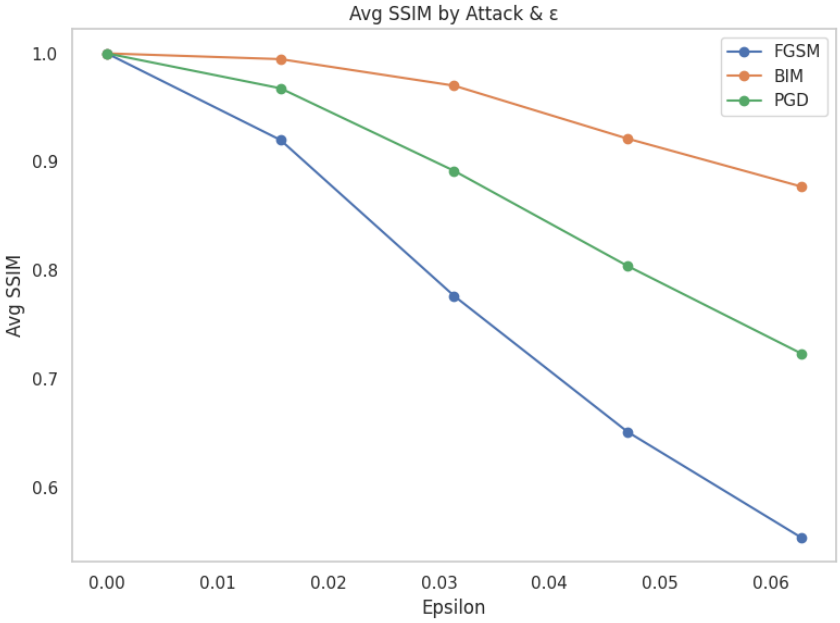

Figure 4.10: Average SSIM between clean and adversarial images, across perturbation levels.

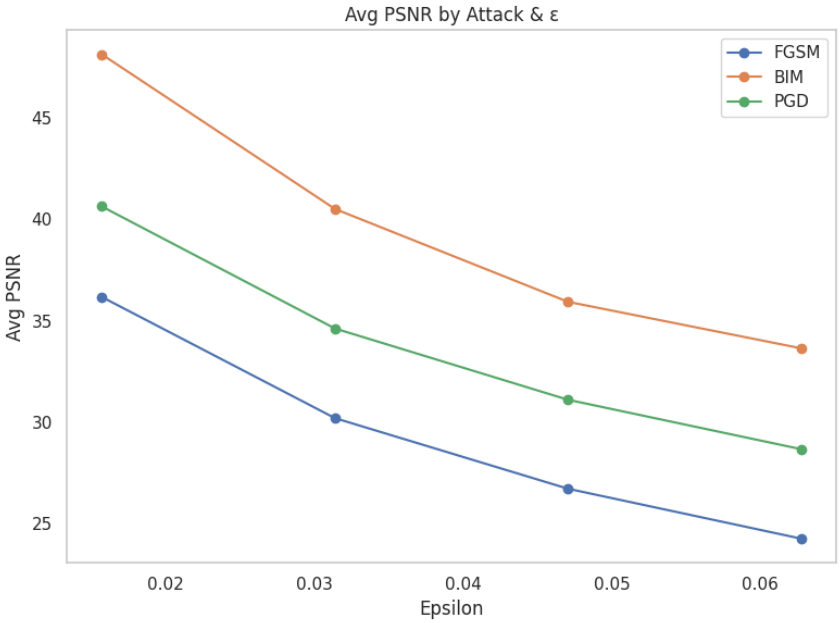

Figure 4.11: Average PSNR between clean and adversarial images, across perturbation levels.

Table 4.10 and Figures 4.10 4.11 summarize the results. FGSM introduces the strongest visual artifacts. At $\epsilon = 0.0157$, it already lowers SSIM to 0.92 and PSNR to 36.2 dB. At $\epsilon = 0.0314$, these values drop sharply to 0.78 and 30.2 dB, and by $\epsilon = 0.0627$, SSIM falls to 0.55 and PSNR to 24.2 dB. This shows that FGSM produces

highly noticeable perturbations. BIM is much more subtle. At $\epsilon = 0.0157$, SSIM remains extremely high (0.99) and PSNR is 48.2 dB, indicating that perturbations are almost imperceptible. Even at $\epsilon = 0.0627$, SSIM stays above 0.87 and PSNR at 33.6 dB, which is still relatively high quality. PGD lies between FGSM and BIM. At $\epsilon = 0.0157$, its SSIM is 0.97 and PSNR 40.6 dB. As ϵ increases, PGD's perturbations become more visible: SSIM drops to 0.72 and PSNR to 28.7 dB at $\epsilon = 0.0627$.

Conclusion. FGSM sacrifices perceptual quality most heavily, while BIM maintains the highest similarity to the clean image. PGD strikes a middle ground: its perturbations are noticeable but less extreme than FGSM. This shows that iterative attacks can fool the model while remaining stealthier in terms of human perception.

Table 4.10: Perceptual similarity of adversarial images (average SSIM and PSNR).

Attack	ϵ	Avg. SSIM	Avg. PSNR (dB)
FGSM	0.0157	0.9203	36.17
	0.0314	0.7767	30.19
	0.0471	0.6514	26.70
	0.0627	0.5541	24.24
BIM	0.0157	0.9947	48.16
	0.0314	0.9703	40.52
	0.0471	0.9214	35.93
	0.0627	0.8774	33.64
PGD	0.0157	0.9678	40.64
	0.0314	0.8918	34.62
	0.0471	0.8042	31.10
	0.0627	0.7236	28.66

4.6.1 Comparison and Summary

FGSM sacrifices perceptual quality most severely, while BIM produces adversarial examples that are almost indistinguishable from the original images at small ϵ values. PGD provides a compromise, balancing attack success with moderate levels of distortion. These results highlight that iterative attacks are not only more effective at fooling the model but also more stealthy to human observers compared to one-step methods like FGSM.

The evaluation highlights both the vulnerabilities and systematic patterns across attacks and class frequencies. In the next chapter, we interpret these findings in light of the research questions and discuss their implications for robustness evaluation.

5 Chapter 5: Discussion & Outlook

Chapter overview

Building directly on the evaluation results from Chapter 4, this chapter interprets the findings and answers the three guiding research questions introduced in Chapter 1. It also considers limitations and outlines directions for future work.
This final chapter interprets the results, discusses limitations, and outlines potential directions for future work.

5.1 Answer to RQ1: Bucket Sensitivity

Research Question 1. Does adversarial robustness vary systematically across class-frequency strata (Rare, Medium, Frequent), or is it largely independent of how often a class appears in the training data? That is: Do models behave differently in terms of robustness when a class appears rarely versus frequently, or does the number of times a class is seen during training have no real effect on how vulnerable it is to adversarial attacks?

To address this question, we analyze the evaluation results from Chapter 4 stratified by class frequency. The following subsections examine how different attacks affect Rare, Medium, and Frequent classes in terms of accuracy, confidence, and overall robustness. This step-by-step analysis provides the basis for answering RQ1.

5.1.1 Analysis of Bucket Sensitivity

5.1.1.1 FGSM shows frequency sensitivity

FGSM results reveal a modest dependence on class frequency. At $\epsilon = 1/255$, Accuracy Drop is ≈ 0.76 for Frequent classes, 0.74 for Medium, and 0.71 for Rare (Section 4.1.1). Relative Accuracy Drop confirms this pattern, saturating around 0.90

with Frequent classes consistently suffering the largest degradation (Section 4.2.1). Thus, FGSM tends to affect Frequent classes more strongly, especially at small perturbation sizes.

This frequency sensitivity, however, is specific to FGSM's one-step nature. To see whether such effects persist under stronger attacks, we next examine BIM and PGD.

5.1.1.2 Iterative attacks erase differences

BIM and PGD rapidly reduce accuracy across all buckets. By $\epsilon = 8/255$, adversarial accuracy falls to zero in every bucket, so Accuracy Drop equals the clean accuracy (0.88 for Frequent/Medium, 0.81 for Rare). Relative Accuracy Drop reaches 1.0 across all buckets (Sections 4.2.2–4.2.3). This shows that iterative optimization removes any frequency-dependent gap.

Since iterative attacks erase frequency gaps in accuracy, it is natural to ask whether frequency still plays a role in terms of confidence. For this, we turn to the Confidence Drop metric.

5.1.1.3 Confidence drops are broadly uniform

Confidence Drop values are highly consistent across buckets. FGSM reduces confidence by 0.62 (Frequent), 0.59 (Medium), and 0.55 (Rare). BIM and PGD show slightly larger but very similar reductions: 0.66, 0.63, and 0.58 respectively (Section 4.3). This indicates that adversarial perturbations universally erode certainty, regardless of frequency.

With confidence reductions also proving uniform, one might question whether frequency-based stratification is useful at all. The following subsection addresses this point.

5.1.1.4 Value of Frequency-Based Stratification

Even though frequency effects are modest overall, stratifying results by frequency proved valuable. It exposed subtle gaps (e.g., FGSM sensitivity to Frequent classes) that would be hidden if only global averages were reported. This stratification also adds a fairness-aware perspective to robustness evaluation (Section 3.5).

Overall, while frequency-based stratification reveals only modest effects, its value lies in exposing subtle patterns that would remain invisible in global averages. This prepares the ground for the consolidated answer to RQ1.

5.1.2 Answer to RQ1

The analysis of frequency strata shows that adversarial robustness does *not* vary systematically with how often a class appears in the training data. FGSM revealed a modest dependence: Frequent classes showed slightly larger accuracy and relative accuracy drops at small perturbation budgets. However, this effect vanished under iterative attacks. Both BIM and PGD quickly reduced performance across all buckets to the same level, erasing any differences between Rare, Medium, and Frequent classes. Confidence Drop results reinforce this picture: across all three buckets, confidence declined by similar amounts, with no systematic advantage for less frequent classes. Taken together, these findings indicate that data frequency is not a decisive factor for adversarial robustness. Subtle effects exist in the one-step FGSM case, but they disappear under stronger attacks. Thus, robustness outcomes are largely independent of class frequency, and *attack type emerges as the dominant driver of vulnerability.*

In summary, the evidence shows that class frequency does not systematically determine robustness: apart from a modest sensitivity under FGSM, iterative attacks (BIM, PGD) eliminate these differences and reduce performance uniformly across all strata. This finding implies that *attack type*, rather than data frequency, is the key driver of adversarial vulnerability.
Having clarified the limited role of class frequency, the logical next step is to examine the attacks themselves in more detail. RQ2 therefore shifts the perspective from "which classes are more or less affected" to "how do different attack methods compare." By contrasting FGSM, BIM, and PGD along dimensions of effectiveness (accuracy, confidence, ASR) and stealth (perturbation size, SSIM, PSNR), we gain a direct understanding of their relative strengths and weaknesses.

5.2 Answer to RQ2: Attack Comparison

Research Question 2. At matched budgets, how do FGSM, BIM, and PGD differ in effectiveness (accuracy degradation, confidence degradation, attack success rate) and stealth (perturbation size, perceptual similarity)?

To answer this question, we draw directly on the evaluation results from Chapter 4. The analysis begins by comparing attack success rates, then examines perturbation size in ℓ_2 norm, perceptual similarity (SSIM and PSNR), and finally confidence degradation. Together, these metrics allow us to contrast FGSM, BIM, and PGD in terms of both effectiveness and stealth.

5.2.1 Analysis of Attack Differences

5.2.1.1 Attack success rates

As reported in Section 4.4, FGSM achieves an ASR of 92% at $\epsilon = 0.0157$ but plateaus below 90% at larger budgets. By contrast, BIM and PGD surpass 97% at $\epsilon = 0.0157$ and achieve 100% success from $\epsilon = 0.0314$ onward. This demonstrates that iterative attacks are consistently stronger and more reliable than FGSM.

High success rates alone, however, do not tell us how costly the attacks are in terms of distortion. To assess efficiency, we next examine perturbation size in the ℓ_2 norm.

5.2.1.2 Perturbation size in ℓ_2 norm

Section 4.5 shows that FGSM produces the largest perturbations: its ℓ_2 norm grows from 6.0 at $\epsilon = 0.0157$ to 23.8 at $\epsilon = 0.0627$. BIM achieves the same attack success with much smaller perturbations ($\ell_2 = 1.5$ to 8.1 across the same range). PGD lies between the two (3.6 to 14.3). Thus, BIM is the most efficient in terms of distortion magnitude.

While ℓ_2 captures pixel-space magnitude, it does not reflect how natural or visible the perturbations appear to humans. This motivates a comparison using perceptual similarity metrics (SSIM, PSNR).

5.2.1.3 Perceptual similarity (SSIM and PSNR)

As shown in Section 4.6, FGSM causes the strongest perceptual degradation. At $\epsilon = 0.0314$, its SSIM drops to 0.78 and PSNR to 30.2 dB, making perturbations visibly noticeable. BIM is much more subtle: SSIM remains at 0.97 and PSNR at 40.5 dB at the same budget, indicating nearly imperceptible changes. PGD lies between FGSM and BIM, with SSIM = 0.89 and PSNR = 34.6 dB.

Visual similarity, however, is only part of the story. Even when an image still looks natural, the model's confidence may already be collapsing. We therefore turn to Confidence Drop as a complementary signal.

5.2.1.4 Confidence degradation

All attacks cause similar confidence declines (Section 4.3), with average drops of 0.55–0.66 across buckets. Iterative attacks reduce confidence slightly more than FGSM, but the overall effect is consistent: adversarial noise always erodes certainty, even when accuracy is preserved.

5.2.2 Answer to RQ2

The comparison of FGSM, BIM, and PGD demonstrates clear differences in both effectiveness and stealth at matched perturbation budgets. FGSM reached high success rates quickly but required relatively large perturbations in ℓ_2 norm and produced visible artifacts, as reflected in low SSIM and PSNR values. BIM, in contrast, achieved the same or higher success with much smaller ℓ_2 distortions and nearly imperceptible perturbations, making it the most efficient and stealthy method. PGD consistently matched BIM's success rate, while producing distortions that were moderate in both size and perceptual visibility. Across all three methods, confidence declined in similar amounts, showing that erosion of certainty is a universal effect of adversarial noise.
These results make clear that attack type matters greatly: FGSM is fast but inefficient, BIM is subtle and highly effective, and PGD offers a balance between the two. Thus, iterative methods (BIM and PGD) clearly outperform FGSM, with PGD providing the best overall trade-off between strength and stealth.

In short, attack type determines not only whether adversarial examples succeed but

also how visible or efficient those perturbations are. Having compared FGSM, BIM, and PGD directly, we now turn to a broader question: how does increasing attack strength affect the balance between model degradation and perceptual stealth? This trade-off is addressed in RQ3.

5.3 Answer to RQ3: Strength–Stealth Trade-off

Research Question 3. How does increasing attack strength shift the balance between model degradation and perceptual similarity, measured via ℓ_2, SSIM, and PSNR? That is: If we let the attacker change the image more strongly, how does that affect the trade-off between fooling the model and still keeping the image looking natural?

To address this question, we analyze how model performance and perceptual quality change as the perturbation budget ϵ increases. The following subsections first examine model degradation in terms of accuracy and attack success rate, then quantify distortion growth with ℓ_2, SSIM, and PSNR, and finally compare how FGSM, BIM, and PGD differ in balancing strength and stealth. This stepwise analysis provides the basis for answering RQ3.

5.3.1 Analysis of Strength–Stealth Trade-offs

5.3.1.1 Model degradation grows with ϵ

As shown in Sections 4.1 and 4.4, larger perturbation budgets (ϵ) always lead to stronger model degradation. Accuracy drops reach 0.8–0.9 by $\epsilon = 4/255$ under all attacks, and ASR climbs to 100% for BIM and PGD at $\epsilon \geq 0.0314$. Thus, higher ϵ consistently strengthens attacks.

While these results confirm that larger perturbation budgets consistently make attacks more powerful in terms of accuracy and success rate, they do not yet reveal the cost of this increased strength. To capture how much the input itself is altered, we next examine distortion metrics in pixel and perceptual space.

5.3.1.2 Distortion metrics also grow with ϵ

Section 4.5 shows that ℓ_2 perturbation norms increase linearly with ϵ. FGSM produces the largest distortions (23.8 at $\epsilon = 0.0627$), BIM the smallest (8.1), and PGD lies in between (14.3). Similarly, perceptual quality (Section 4.6) declines steadily: for FGSM, SSIM falls from 0.92 to 0.55 and PSNR from 36.2 dB to 24.2 dB as ϵ increases, while BIM retains higher values (SSIM 0.99→0.88, PSNR 48.2→33.6 dB). PGD again lies between these extremes.

These trends already suggest that different attack methods achieve their strength at very different perceptual costs. To make this contrast explicit, we now compare FGSM, BIM, and PGD directly in terms of how they balance attack effectiveness against stealth.

5.3.1.3 Differences between attack types

- FGSM: maximizes strength quickly but sacrifices stealth, producing large and visible perturbations.
- BIM: achieves maximum strength with minimal distortion, making it the most stealthy.
- PGD: combines both; it reaches full strength while keeping distortions moderate.

Together, these comparisons highlight that the relationship between strength and stealth is not universal but strongly method-dependent: FGSM favors raw power, BIM emphasizes subtlety, and PGD balances the two. This sets the stage for the consolidated answer to RQ3.

5.3.2 Answer to RQ3

The analysis of different perturbation budgets shows a consistent trade-off between attack strength and perceptual stealth. As ϵ increases, model degradation becomes more severe: accuracy drops approach their maximum and attack success rates reach 100% for BIM and PGD. At the same time, distortion metrics also grow: ℓ_2 norms increase linearly with ϵ, and perceptual quality measured by SSIM and PSNR declines steadily. FGSM demonstrates this trade-off most clearly, producing strong model degradation but also highly visible perturbations. BIM, on the other hand,

achieves full attack success with minimal and often imperceptible changes, while PGD occupies a middle ground with moderate levels of distortion.
Together, these findings confirm that stronger attacks inevitably reduce perceptual similarity, but the rate of this trade-off depends on the attack method. FGSM sacrifices stealth for strength, BIM shows that attacks can be both strong and subtle, and PGD balances both dimensions. Therefore, robustness evaluation must jointly consider not only effectiveness but also perceptual plausibility in order to capture the true cost of adversarial attacks.

5.4 Summary of the results

This chapter explicitly addressed the three guiding research questions of this thesis. The findings are summarized below:

- **RQ1: Bucket Sensitivity.** We asked whether adversarial robustness varies systematically across class-frequency strata (Rare, Medium, Frequent), or whether it is largely independent of class frequency. Our results show that FGSM exhibits modest sensitivity, with Frequent classes degrading slightly more, but BIM and PGD erase these differences and drive all buckets to complete failure. Confidence drops are consistent across buckets. Thus, robustness is largely independent of class frequency, and attack type is the dominant factor.

- **RQ2: Attack Comparison.** FGSM is fast but inefficient, producing large and visible perturbations. BIM is precise and stealthy, achieving 100% success with minimal distortion. PGD balances both strength and stealth, making it the strongest overall. Therefore, iterative methods (BIM, PGD) clearly outperform FGSM in both effectiveness and subtlety.

- **RQ3: Strength–Stealth Trade-off.** Increasing perturbation size (ϵ) always strengthens attacks (higher accuracy drop, higher ASR) but reduces perceptual similarity (larger ℓ_2, lower SSIM/PSNR). FGSM demonstrates this trade-off most severely, while BIM shows that strong attacks can remain stealthy. PGD provides a middle ground. Hence, robustness evaluation must consider not just strength but also perceptual stealth.

List of Figures

Bibliography

[BMM18] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. "A Systematic Study of the Class Imbalance Problem in Convolutional Neural Networks". In: *arXiv:1710.05381* (2018).

[CH20] Francesco Croce and Matthias Hein. "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks". In: *arXiv preprint arXiv:2003.01690* (2020).

[Cro+21] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, and et al. "RobustBench: a standardized adversarial robustness benchmark". In: *NeurIPS Datasets and Benchmarks*. 2021.

[CW17] Nicholas Carlini and David Wagner. "Towards Evaluating the Robustness of Neural Networks". In: *IEEE Symposium on Security and Privacy (S&P)*. 2017.

[Eyk+18] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. "Robust Physical-World Attacks on Deep Learning Visual Classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.

[Fin+19] Samuel G. Finlayson, John D. Bowers, Joichi Ito, Jonathan L. Zittrain, Andrew L. Beam, and Isaac S. Kohane. "Adversarial Attacks Against Medical Deep Learning Systems". In: *Science* 363.6433 (2019), pp. 1287–1289.

[GSS15] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and Harnessing Adversarial Examples". In: *arXiv preprint arXiv:1412.6572* (2015).

[Guo+17] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. "On Calibration of Modern Neural Networks". In: *arXiv:1706.04599* (2017).

[HG17] Dan Hendrycks and Kevin Gimpel. “A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks”. In: *arXiv preprint arXiv:1610.02136* (2017).

[HZ10] Alkaram B. Hore and Djemel Ziou. “Image Quality Metrics: PSNR vs. SSIM”. In: *ICPR Workshops*. Open PDF widely available. 2010.

[KGB17a] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. “Adversarial Examples in the Physical World”. In: *arXiv preprint arXiv:1607.02533* (2017).

[KGB17b] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. “Adversarial Machine Learning at Scale”. In: *International Conference on Learning Representations (ICLR) Workshop*. Open preprint: arXiv:1611.01236. 2017.

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning”. In: *Nature* 521.7553 (2015), pp. 436–444.

[Mad+18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. “Towards Deep Learning Models Resistant to Adversarial Attacks”. In: *International Conference on Learning Representations (ICLR)*. 2018.

[NJ02] Andrew Y. Ng and Michael I. Jordan. “On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes”. In: *Advances in Neural Information Processing Systems (NeurIPS) 14*. 2002.

[Pap+17] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. “Practical Black-Box Attacks against Machine Learning”. In: *arXiv preprint arXiv:1602.02697* (2017).

[Rus+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. “ImageNet Large Scale Visual Recognition Challenge”. In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252.

[Sha+16] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. “Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition”. In: *ACM Conference on Computer and Communications Security (CCS)*. 2016.

[Sze+14] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. “Intriguing properties of neural networks”. In: *arXiv preprint arXiv:1312.6199* (2014).

[Wan+04] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. “Image Quality Assessment: From Error Visibility to Structural Similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612.

Declaration on the use of tools and AI-assisted writing tools

I confirm that when using IT-/AI-assisted writing tools, I have fully listed these tools in the following overview of the aids used, including their product name and my source of access, and/or have marked the relevant text passages in the thesis as written with IT/AI-generated support.

I am aware that deception or attempted deception will be sanctioned according to the examination regulations applicable to me.

The following aids and AI-assisted writing tools were used in the preparation of this thesis:

- DeepL Write, — I used it to correct grammar and vocabulary errors ; **Provider / Source (Bezugsquelle):** DeepL SE ; accessed via **Website:** `https://www.deepl.com/write`

Eidesstattliche Versicherung

(Affidavit)

Alain Yvan Ngeukeu Ngongang
Name, Vorname
(surname, first name)

231847
Matrikelnummer
(student ID number)

☒ Bachelorarbeit
(Bachelor's thesis)

☐ Masterarbeit
(Master's thesis)

Titel
(Title)

Comparitive Analysis of Adversarial Methods

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem oben genannten Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

I declare in lieu of oath that I have completed the present thesis with the above-mentioned title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution before.

Bochum, 03/09/2025
Ort, Datum
(place, date)

Unterschrift
(signature)

Belehrung:
Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin") zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:
Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to EUR 50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the Chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, Section 63 (5) North Rhine-Westphalia Higher Education Act (*Hochschulgesetz, HG*).

The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund University will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:*

Bochum, 03/09/2025
Ort, Datum
(place, date)

Unterschrift
(signature)

*Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.