Balancing Image Quality and Attack Effectiveness in Multi-Objective Adversarial Image Generation

José Areia
jose.a.areia@ipleiria.pt
Polytechnic University of Leiria
Leiria, Portugal

Leonel Santos
leonel.santos@ipleiria.pt
Polytechnic University of Leiria
Leiria, Portugal

Rogério Luís de C. Costa
rogerio.l.costa@ipleiria.pt
Polytechnic University of Leiria
Leiria, Portugal

Abstract

Adversarial attacks pose a significant threat to deep neural networks (DNNs) in computer vision by introducing subtle perturbations that can mislead these models. This paper proposes a multi-objective generative adversarial network (GAN), enhanced with an encoder, to generate adversarial images capable of deceiving DNNs. The model was trained on images perturbed by four distinct attacks at varying perturbation levels and tested across five DNN architectures. Performance was evaluated not only by fooling rate (FR) but also by image quality, using the Fréchet Inception Distance (FID) and Learned Perceptual Image Patch Similarity (LPIPS) metrics. The generated adversarial images achieved a FR of up to 89.63% while maintaining high image quality, with LPIPS and FID scores as low as 0.23 and 25, respectively. These results demonstrate that the proposed approach effectively balances deception and image fidelity, supporting the feasibility of this multi-objective method.

CCS Concepts

• **Computing methodologies** → **Machine learning**; • **Security and privacy** → *Software and application security*.

Keywords

Adversarial Attacks, Deep Neural Networks, Generative Adversarial Networks, Perturbation-based Attacks.

1 Introduction

Deep neural networks (DNNs) have achieved remarkable success in computer vision tasks [1, 2]. However, they remain susceptible to adversarial examples—subtle input perturbations that can significantly degrade the prediction accuracy [3–5]. This vulnerability is especially critical in image recognition, where small changes to input images can cause misclassifications, thereby raising concerns about the robustness of DNN-based systems [2, 6]. Prior research has explored the generation of adversarial perturbations that successfully mislead models with different network architectures [7, 8]. Nevertheless, such perturbations are often tailored to specific images, limiting their generalisability [9, 10]. To overcome this limitation, universal adversarial perturbations (UAPs) have been proposed [11]. UAPs aim to generate a single perturbation capable of misleading multiple DNNs across diverse inputs, thus improving their practicality for evaluating model robustness in real-world scenarios [12, 13].

KDD '25, Toronto, Canada
2025. ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Generative models, such as Generative Adversarial Networks (GANs) [14] and their extensions [15–18], are capable of synthesising high-fidelity images and capturing complex visual patterns, including those that may be imperceptible to the human eye. These properties align closely with the characteristics of adversarial perturbations [19–22]. Several studies have explored using generative models within adversarial frameworks to deceive other models. However, relying solely on the fooling rate (FR) as a performance metric may result in adversarial examples with poor visual quality, where perturbations become noticeable to human observers [23].

This study proposes a multi-objective generative model that achieves high fooling rates while preserving image quality. Our method generates transferable adversarial examples that consistently deceive multiple models. We evaluate the model using five perturbation methods from four state-of-the-art attacks on five well-known DNN networks. Additionally, we perform ablation studies to assess attack success and image quality.

Therefore, the main contributions of this work are: *i*) a multi-objective generative model that embeds adversarial characteristics while preserving image fidelity, *ii*) an experimental evaluation of how different types and magnitudes of perturbations affect both the visual quality of adversarial images and the FR, and *iii*) a comprehensive methodology for evaluating the generation of adversarial images with diverse perturbation strategies targeting adversarial networks.

The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 presents the proposed methodology in detail. Section 4 outlines the experimental evaluation and discusses the results. Finally, Section 5 concludes the paper and highlights directions for future research.

2 Related Work

Adversarial attacks. The universe of adversarial attacks can be divided into two categories: instance-specific and universal attack methods. The concept of adversarial examples was first introduced by Szegedy *et al.* [1]. These attacks are designed to create perturbations for specific images using training or substitute data. Common instance-specific attacks, such as the Fast Gradient Sign Method (FGSM) [3], I-FGSM and MI-FGSM [24], Projected Gradient Descent (PGD) [7], and Output Diversified Sampling (ODS) [25], typically utilise gradient information from a well-trained target Convolutional Neural Network (CNN) model (*i.e.*, white-box attack) or a surrogate model (*i.e.*, black-box attack) to perturb images. In contrast, universal attack methods aim to develop a single perturbation that is independent of any specific image. Examples of this approach include Universal Adversarial Perturbation (UAP) [11], Truncated Ratio Maximisation UAP (TRM-UAP) [10], and Stochastic Gradient

Aggregation (SGA) [2]. Applying a single UAP to various benign samples efficiently generates numerous adversarial examples.

Generative adversarial techniques. Generative models can embed adversarial traits that mislead classification systems, either on individual instances or across datasets. Mopuri *et al.* [26] proposed the Network for Adversary Generation (NAG), which uses generative methods to produce diverse, transferable perturbations that fool classifiers. Unlike methods that generate a single perturbation, NAG outputs a range of adversarial examples. Expanding on this, Poursaeed *et al.* [27] introduced Generative Adversarial Perturbations (GAP), capable of producing both universal and image-specific perturbations for classification and segmentation tasks. Their approach achieved over 80% fooling rate on a pre-trained VGG-16 model. Xiao *et al.* [19] later proposed AdvGAN, a generative model uses adversarial loss to produce examples that mislead classifiers. In another study, Sun *et al.* [18] explored vulnerabilities in GAN-based image fusion, showing that even subtle perturbations can severely impact performance, revealing the fragility of such models.

Multi-objective generative models. The concept of multi-objective learning within generative models was first introduced by Durugkar *et al.* [28], who proposed Generative Multi-Adversarial Networks (GMAN). This approach involves training the generator against a softmax-weighted arithmetic average of K different discriminators. The results of their study showed that using a simple average of the discriminators' losses yielded the best performance across most evaluation metrics. Subsequently, Albuquerque *et al.* [29] applied a multiple gradient descent method with hypervolume maximisation, achieving a more effective balance between sample quality and computational cost. In a more application-oriented context.

3 Multi-Objective Adversarial Image Generation

This section presents key concepts for crafting universal and instance specific perturbations to generate adversarial images, then outlines our proposed adversarial generative architecture.

3.1 Preliminary

Universal perturbations. In UAP-based attacks [2, 10, 11], the goal is to deceive a given CNN model f by identifying a universal perturbation δ, derived from a perturbation vector v within a data distribution $\mathbb{R}^d$. This perturbation δ is designed to maximise the classification loss $\mathcal{L}$ (*e.g.*, cross-entropy) and mislead the classifier f on nearly *all data points* sampled from μ, where μ represents the distribution of images in $\mathbb{R}^d$. That is, these attacks aim to:

$$f(x+v) \neq f(x) \text{ for "most" } x \sim \mu, \tag{1}$$

where $f(x)$ represents the output of the model f, *i.e.*, the predicted label for each image $x \in \mathbb{R}^d$. Moreover, it is important to recall that these attacks seek to find v that satisfies a main constraint, as defined in the Equations 2.

$$\|v\|_p \leq \xi\,, \ \mathop{\mathbb{P}}_{x\sim\mu}(f(x+v) \neq f(x)) \geq 1-\delta \tag{2}$$

The hyperparameter ξ controls the magnitude of the perturbation vector v, while δ, representing the universal perturbation, also specifies the desired fooling rate across all images sampled from the distribution.

Instance-specific perturbations. While UAP-based attacks compute a universal perturbation δ applicable to a wide range of images, instance-specific attacks [4, 7, 9, 30] generate a δ tailored to individual inputs. However, both approaches share a common limitation: the perturbation δ must be computed first and then manually applied to each image. At scale, this process becomes computationally intensive—first requiring the calculation of δ for each sample $\mu \in \mathbb{R}^d$, followed by its manual application to the images, all while maintaining the correct perturbation magnitude ξ. Furthermore, these methods may lack robustness in terms of model transferability, as a perturbation crafted to fool a model f may perform poorly when applied to a different model.

3.2 Adversarial Image Generation Workflow

Generative models synthesize images through competitive training, while traditional adversarial attacks apply perturbations directly to existing images. We propose that a trained generative model can learn adversarial features from perturbed data while preserving the original image's appearance through a competitive, multi-objective training process. Once trained, the model can generate perturbed versions of input images that effectively mislead CNN classifiers. To validate this approach, we adopt a five-stage methodology: *i*) data acquisition and preprocessing, *ii*) adversarial sample synthesis, *iii*) training of the generative framework, *iv*) latent space encoder optimisation, and *v*) quantitative and qualitative evaluation of the generated images.

For the first two stages, we utilised a pre-existing dataset and applied a series of adversarial attacks. To assess the impact of each perturbation magnitude ϵ on the pre-trained models, we generated distinct adversarial datasets for each attack, with each dataset corresponding to a specific perturbation level. The resulting adversarial images were then organised by attack type, forming a diverse n-set of perturbed images categorised accordingly. In the third stage, we developed a custom generative architecture based on an existing GAN framework, as detailed in next section.

3.3 Proposed Generative Architecture

The proposed generative architecture, illustrated in Figure 1, consists of two discriminators, denoted as D and D', a generator G, and a classifier C. The discriminator D functions as a standard, or *vanilla*, discriminator—that is, it assesses whether the input image is real or fake, thereby guiding the generator to produce images that closely resemble the originals. In contrast, the discriminator D' is specifically designed to capture adversarial traits from perturbed images and to encourage their subtle integration into the generated outputs in a way that remains imperceptible to the human eye.

The generator G follows an encoder–decoder architecture incorporating instance normalisation throughout. It begins with a single convolutional layer followed by a ReLU activation. This is followed by two downsampling blocks, each consisting of a convolutional layer, instance normalisation, and ReLU activation. The core of the network comprises six residual blocks, each containing two convolutional layers with instance normalisation, connected via skip connections to facilitate information flow and gradient

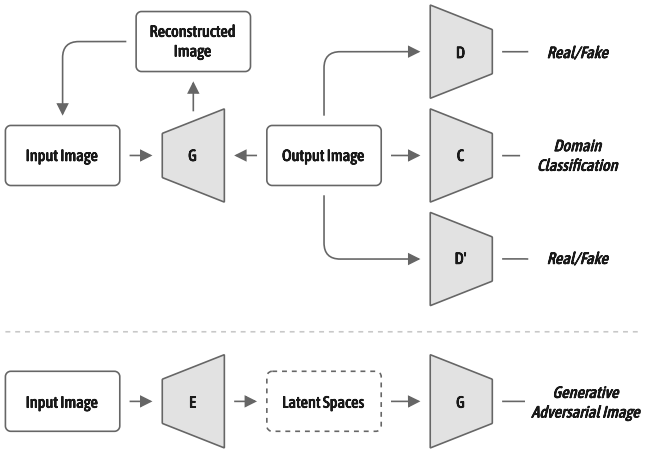

Figure 1: Proposed multi-objective generative architecture. The upper architecture depicts the multi-objective generative model incorporating multiple discriminators. The lower architecture illustrates the encoder training process, which aims to replicate real images through the trained generator.

stability. Following the residual blocks, two upsampling blocks are implemented using transposed convolutions, each paired with instance normalisation and ReLU activations. The final output layer employs a Tanh activation function to generate the output image.

The discriminator D adopts a PatchGAN [17] architecture, consisting of six convolutional layers with spectral normalisation and LeakyReLU activations. The discriminator D' shares a similar convolutional backbone but concludes with a fully connected, spectral normalised linear layer for classification purposes. The auxiliary classifier C mirrors the convolutional structure of the discriminators and terminates with a global convolutional classification head, adapted to the downsampled spatial dimensions of the input image.

It is possible to assign different importance to each discriminator's objective by adjusting the nadir slack and the weights for both D and D'. The nadir slack represents the difference between the nadir point and a reference point, known as the *objective point*. Experimentally, this slack value is used to dynamically update the nadir point during training to improve stability [29], as described by the following equation:

$$z^{nad} = \max_i \ell_i \cdot \varsigma^{nad} + \kappa, \tag{3}$$

where ℓ_i denotes the individual objective losses, and κ is a small constant set to $\kappa = 1\times10^{-8}$. This nadir point z^{nad} was subsequently used during the training process, in conjunction with the losses from the various objectives, to compute the hypervolume. This hypervolume was then used to calculate the loss of the generator G, thereby providing a measure of how effectively the training process covers the objective space towards the ideal point in the context of multi-objective optimisation.

Since our objective is to take an image as input and output the same image with a given perturbation δ, we must replicate this image by extracting its latent space information. To achieve this, we developed an encoder E that learns structured latent representations of real data, which helps the generator produce more realistic and diverse outputs. Therefore, instead of relying on $y = G(z)$ to generate an adversarial image, we use $y = G(E(x))$, where $E(x)$ is the encoder's output when the real image x is input. Thus, in the fourth stage, we proceed with training the encoder. For this training, we utilise both the real images from the dataset and the generator G for each test iteration.

Main algorithm. The Algorithm 1 presents a structured overview of the adversarial training process, including the evaluation procedure. The evaluation metrics used are described in Section 3.4.

Algorithm 1 Adversarial training process with an evaluation procedure.

Require: Surrogate CNN models $f = \{f_1, \ldots, f_5\}$ for evaluation
Input: Original dataset $\mathcal{D}$ and a list of attacks $\mathcal{I}$
Output: JSON file $m = \{m_1, \ldots, m_4\}$
  for $i \in I$, where $\mathcal{I} = \{\text{FGSM, UAP, SGA, TRM}\}$ **do**
    Generate perturbation δ_i
    Create perturbed dataset: $\mathcal{D}_i = \mathcal{D} + \delta_i$
    Train generator G_i with objectives:
      $D(G(x)) \approx D(x)$ ▷ Preserve realism
      $D'(G(x)) \approx D(x_i)$ ▷ Capture adversarial traits
    Train encoder E on $\mathcal{D}$ to extract latent codes z:
      $z = (G(E(x)),\ x \in \mathcal{D}$
    for all $x \in \mathcal{D}$ **do**
      if $\hat{y} = f(x) = y$ by all $f_j \in f$ **then**
        for all $f_j \in f$ **do**
          Compute FR$(f_j, \mathcal{D}_i)$
        end for
      end if
    end for
    Compute FID$(\mathcal{D}, \mathcal{D}_i)$
    Compute LPIPS$(\mathcal{D}, \mathcal{D}_i)$
    $m_i = \{\text{FR}_i,\ \text{LPIPS}_i,\ \text{FID}_i\}$
  end for
  return $m = \{m_1, \ldots, m_4\}$

3.4 Evaluation Metrics

To assess if the generated images can mislead a given CNN model, we first verify the correct classification of original images x and then test their adversarial counterparts x'. This approach identifies potential false positives, ensuring reliable evaluation and yielding a variation of the FR [12, 31]. Since generative models produce images that incorporate features from the reference set but differ significantly from the original images visually, their quality cannot be accurately assessed through a simple pixel-by-pixel comparison with training set images. To address this, we employ two metrics: the Fréchet Inception Distance (FID) [32] and the Learned Perceptual Image Patch Similarity (LPIPS) [33].

The FID metric compares the means and covariances of features from the deepest layer of the Inception v3 network [3], capturing high-level semantic information. It measures the similarity between two image sets using the Fréchet distance between their feature distributions, modelled as multivariate Gaussians [32]:

$$\text{FID} = \left\|\mu_r - \mu_g\right\| + \text{T}_\text{r}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}), \tag{4}$$

where μ_r, μ_g are the means of real and generated image distributions, and Σ_r, Σ_g are their covariance matrices. A lower FID score

indicates greater similarity between generated and real images in terms of feature distributions.

The LPIPS metric measures perceptual similarity between two images using features from a pre-trained deep neural network, unlike FID which compares image sets [33]. It calculates a weighted distance between the deep features of the images. Feature maps F_i and F_j from layer l for images I_i and I_j are normalised:

$$\hat{F}_{ij} = \frac{F_{ij}}{||F_{ij}||_2} \tag{5}$$

Once normalised, the distance between the two images is computed as:

$$d^l = \frac{1}{H_l W_l} \sum_{h,w} \left\| \hat{F}^l_{i,hw} - \hat{F}^l_{j,hw} \right\|_2^2, \tag{6}$$

where H_l and W_l represent the spatial dimensions of the feature maps at layer l. The final LPIPS score is a weighted sum of distances across all layers:

$$\text{LPIPS}(I_i, J_i) = \sum_l w_l \cdot d^l \tag{7}$$

Weights w_l are learned from human perceptual judgments, aligning LPIPS more closely with human visual perception than traditional metrics.

4 Experimental Evaluation

In this section, we delve into the experimental aspects of our research. We begin by outlining the experimental setup, followed by an explanation of the experiments conducted. After this, we present the results obtained and discuss our findings.

4.1 Experimental Setup

Dataset. Following [2, 10, 12, 31], we used a publicly available subset of the ImageNet dataset [34], known as Imagewoof [35]. Imagewoof comprises images of 10 different dog breeds. Approximately 10 000 images were used for training, with around 5 000 images allocated for validation. Various modifications were applied to the images to enhance the available data. These included resizing to a specified dimension, random horizontal flipping, random conversion to greyscale, and normalisation using the mean and standard deviation calculated across the three RGB channels. These transformations were implemented using PyTorch's data augmentation utilities. Table 1 presents the parameters associated with each transformation.

Table 1: Image transformations with associated values.

Transformations	Values Applied
Resizing	$W = 244$ x $H = 244$
Random horizontal flips	1 ($True$)
Random greyscale	$P = 0.1$, where $P \Rightarrow Probability$
Mean normalisation	$R = 0.485$, $G = 0.456$, $B = 0.406$
Standard deviation normalisation	$R = 0.229$, $G = 0.224$, $B = 0.225$

Attacks. The proposed generative model is trained not on original images, but on perturbed ones. This enables the model to learn and incorporate adversarial characteristics present in the modified images. To meet this requirement, four adversarial attacks were applied, and the resulting perturbations, denoted by δ, were extracted across five tests, each corresponding to a different perturbation magnitude, $\epsilon \in [0.01, 0.05, 0.10, 0.15, 0.20]$. The attacks used include FGSM [4], UAP [11], and two UAP-based variants: TRM-UAP [10] and SGA [2]. TRM-UAP aims to maximise CNN activations in a fully data-free setting using a truncated ratio maximisation approach, while SGA addresses gradient instability by aggregating multi-step noisy forward gradients and applying a single-step quantised update at each iteration.

The application of δ to an image varies with the attack method used. For both FGSM and UAP attacks, the precomputed δ is applied iteratively to each image, scaled by ϵ to achieve the desired adversarial effect. In the TRM-UAP attack, δ is initially trimmed using a clamping operation:

$$\delta' = min(max(\delta, \epsilon_{min}), \epsilon_{max}), \tag{8}$$

$$\text{where } \epsilon_{min} = -10/255, \epsilon_{max} = 10/255 \tag{9}$$

Next, it was added to each individual batch of images:

$$y = \delta' + \alpha \cdot x, \text{where } \alpha = 1 \tag{10}$$

Finally, a new clamping operation was applied:

$$y' = min(max(y, \epsilon_{min}), \epsilon_{max}), \tag{11}$$

$$\text{where } \epsilon_{min} = 0, \epsilon_{max} = 1 \tag{12}$$

In the SGA attack, the perturbation δ was clamped alongside each batch of images using a minimum value of $min = 0$ and a maximum value of $max = 1$, resulting in the generation of adversarial images denoted as y.

Network architecture. Our network architecture is based on the state-of-the-art GAN, SuperstarGAN [16], with several key modifications. SuperstarGAN typically comprises a generator (G), a discriminator (D), and a classifier (C). Our goal is to generate a custom image resembling the original while incorporating adversarial traits. To achieve this, we introduce a second discriminator (D') to extract adversarial traits during training, along with a custom encoder (E).

The G starts with a 7×7 convolutional layer, followed by downsampling with two 4×4 convolutions, each doubling the feature maps. It uses 6 residual blocks as a bottleneck and upsamples with two transposed convolutions, reducing feature maps to generate a 3-channel image with Tanh activation. The input is an image concatenated with domain label information.

The D includes several convolutional layers with spectral normalisation, starting with a 4×4 convolution followed by LeakyReLU activation. It applies 6 convolutional layers, each doubling the feature maps, and ends with a 3×3 convolution outputting a real/fake score. The D' is designed for adversarial training, starting similarly to D but ending with a fully connected layer classifying the input into one of the labels ($n = 10$).

The C consists of convolutional layers with LeakyReLU activation, starting with a 4×4 convolution and applying 6 downsampling layers. The final layer outputs a class-dimensional vector representing class scores. The E consists of convolutional layers with ReLU activation, beginning with a 4×4 convolution and applying 6 downsampling layers. A final convolution with an adaptive kernel

size outputs a class-dimensional vector representing the inferred domain vector.

Hyperparameters. To generate adversarial images and facilitate comparison using other image quality-based evaluation metrics, we adopted the same hyperparameters used in previous generative-based studies [16, 36–38], with several modifications tailored to the specific requirements of this work. The batch size was set to 128, and the image dimensions were reduced to 128×128. The number of channels was fixed at 3, corresponding to the RGB colour space. The size of the feature maps in G, D, D', C, and E was set to 32. The Adam optimiser [39] was used with a learning rate lr and two momentum parameters: β_1 and β_2. Separate Adam optimisers were employed for G, D, D', and C. The lr for all components was initialised at 2×10^{-4}. For G, D, and D', β_1 was set to 0.0 and β_2 to 0.999. In contrast, for C, β_1 was set to 0.9 and β_2 to 0.999. It is important to note that the lr for each component decayed linearly towards zero over the final 1×10^5 iterations. For the multi-objective learning setup, we used $\varsigma^{nad} = 1.1$, and $\lambda_D = 0.65$ for D and $\lambda_{D'} = 0.35$ for D'.

The training process was conducted for a total of 100 000 epochs. This number was carefully selected based on extensive experimentation. Specifically, we monitored and evaluated the loss values of G, D, D', and C during training with the FGSM attack at $\epsilon = 0.10$.

Evaluation models and metrics. We employed the FR method proposed by data-free universal methods [12, 31], along with a modified variation (see Section 3.2), to assess false positives in our results. Additionally, the FID [32] and LPIPS [33] metrics were used to evaluate the similarity between the generated images and the original ones. Additionally, we evaluate five widely used models, namely AlexNet [40], VGG-16 and VGG-19 [41], ResNet18 and ResNet152 [36].

Setup. All experiments were conducted using Python 3.12 with PyTorch 2.6. Additionally, manual verification was employed to monitor the training process and analyse loss behaviour. The study was carried out on a workstation equipped with a single NVIDIA GeForce RTX 4080 Super GPU, utilising CUDA 12.4.

4.2 Adversarial Network Performance

The results, summarised for three metrics—FR, FID, and LPIPS—across four attacks and five evaluated models, are presented in Table 2. The values represent the average scores over all tested perturbation magnitudes, along with their standard deviations. Note that a higher FR (%) reflects greater attack effectiveness, while lower LPIPS and FID values indicate better perceptual image quality.

Regarding the FR, the overall results range from approximately 66.0% to 84.2%. ResNet18 achieved the highest average FR at 80.7%, while VGG19 yielded the lowest performance, with an average FR of 73.3%.Despite some variation in attack transferability across models, the SGA notably maintains a consistent FR with an average fluctuation of only 1%, while other attacks show slightly larger variations but achieve results approximately 5% higher than SGA.

Across all tests, the average FID scores are similar for all attacks and models, ranging from 32.6 to 37.4. Notably, SGA achieves the lowest—*best*—average score of 34.4, followed by FGSM at 35.7, UAP at 36.1, and TRM-UAP at 36.3. Similarly, the LPIPS scores show little variation across all attacks and methods, ranging from 0.24 to 0.31. Again, SGA performs best with the lowest average of 0.24, followed by TRM-UAP at 0.25, UAP at 0.26, and FGSM with the highest—*worst*—score of 0.30.

Table 2: Summary of performance metrics for the proposed adversarial generative network.

Attack	AlexNet	ResNet152	ResNet18	VGG-16	VGG19
Fooling Rate (FR)					
FGSM	74.5±1.5	80.3±4.7	81.1±3.6	72.0±2.6	70.4±2.6
UAP	70.3±4.2	66.0±3.1	81.6±3.4	72.1±4.7	75.3±4.8
TRM-UAP	76.3±5.0	82.9±2.9	84.2±4.0	74.5±6.5	73.3±6.4
SGA	78.3±9.0	77.9±5.9	77.6±5.7	77.7±7.1	75.5±6.0
Fréchet Inception Distance (FID)					
FGSM	36.6±12.0	33.7±8.2	36.1±10.1	36.4±7.6	35.9±10.9
UAP	36.9±8.5	33.9±6.4	37.2±10.8	35.4±9.6	37.2±9.8
TRM-UAP	36.5±8.1	34.6±7.8	36.7±11.4	37.4±9.9	36.5±10.8
SGA	32.6±7.8	34.9±10.1	36.4±8.8	34.5±10.4	34.0±9.0
Learned Perceptual Image Patch Similarity (LPIPS)					
FGSM	0.30±0.09	0.29±0.10	0.30±0.10	0.31±0.09	0.31±0.09
UAP	0.27±0.06	0.27±0.06	0.26±0.06	0.26±0.06	0.26±0.06
TRM-UAP	0.25±0.05	0.26±0.06	0.25±0.04	0.26±0.05	0.25±0.05
SGA	0.25±0.05	0.24±0.05	0.24±0.05	0.25±0.05	0.25±0.05

Figure 2 illustrates the correlation between the average evaluation metrics across all tested models and attacks. Please note that the FID scores were normalised to the $[0, 1]$ range by dividing the average by 100.

All attacks exhibit a similar trend: as the perturbation magnitude increases, the FR generally increases as well. A minor exception occurs with FGSM at $\epsilon = 0.10$ and $\epsilon = 0.15$, where the FR slightly drops below that at $\epsilon = 0.05$. Similarly, FID and LPIPS values rise with increased perturbation magnitude, indicating a decline in image quality—an expected outcome, as stronger perturbations typically degrade visual fidelity.

4.3 Influence of Discriminator Weight Variation

We further investigate the impact of varying the discriminator weights through ablation testing. Specifically, we set the weights of D and D' to zero separately, rendering each discriminator inactive and allowing us to assess its individual contribution to the model's performance. Experiments were conducted using the baseline configuration described in Section 4.1, where the default values are $\lambda_D = 0.65$ and $\lambda_{D'} = 0.35$. In contrast, the *SuperstarGAN Vanilla* configuration sets $\lambda_D = 1.0$ and $\lambda_{D'} = 0.0$, relying solely on the standard discriminator. Conversely, the *SuperstarGAN Adversarial Only* setup uses $\lambda_D = 0.0$ and $\lambda_{D'} = 1.0$, focusing entirely on adversarial traits. We also compare results with a standard adversarial DCGAN without multi-objective optimisation, using the SGA attack ($\epsilon = 0.10$) on ResNet18. Results are shown in Table 3.

Regarding the FR, the Adversarial DCGAN achieves the highest score at 91%. However, its image quality metrics are significantly worse than the other configurations, suggesting that the resulting images are of poor quality—therefore making misclassification

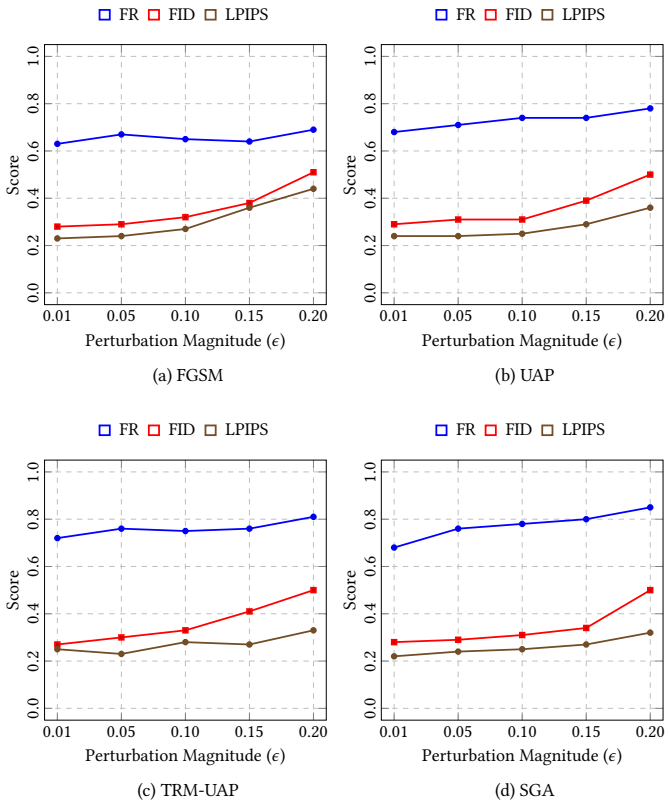

Figure 2: Correlation between the average evaluation metrics—FR, FID, and LPIPS—and the perturbation magnitude of the generated images for each attack, across the tested models: AlexNet, VGG-16, VGG-19, ResNet18, and ResNet152. FID values were normalised to [0,1] by dividing by 100.

more likely due to degradation rather than subtle adversarial manipulation. The *SuperstarGAN Vanilla* setup, by contrast, yields a low FR of 37.5% but demonstrates strong image fidelity, with LPIPS and FID scores of 0.15 and 23.7, respectively. The *SuperstarGAN Adversarial Only* configuration shows a high FR of 89.7%, but again at the cost of quality, reflected in poor LPIPS and FID scores of 0.43 and 52.9. Finally, the our approach offers the best compromise, attaining a robust FR of 80.9% while maintaining acceptable image quality, with LPIPS and FID values of 0.24 and 36.6, respectively.

Table 3: Impact of discriminator weight variation on FR, LPIPS, and FID, in comparison with other architectures. The results are obtained using the SGA attack with a perturbation magnitude of 0.10, targeting the ResNet18 model.

Architecture	FR	LPIPS	FID
Adversarial DCGAN [23]	91.0	0.64	271.0
SuperstarGAN (*Vanilla*)	37.5	0.15	23.7
SuperstarGAN (*Adversarial Only*)	89.7	0.43	52.9
Our Proposal	80.9	0.24	36.6

4.4 Evaluating Adversarial Image Quality

After analysing the results and computing the evaluation metrics, a qualitative assessment—*i.e.*, real-world image inference—was deemed necessary to further validate the effectiveness and visual realism of the generated adversarial samples. Figure 3 illustrates a comparative analysis of real images versus generated images.

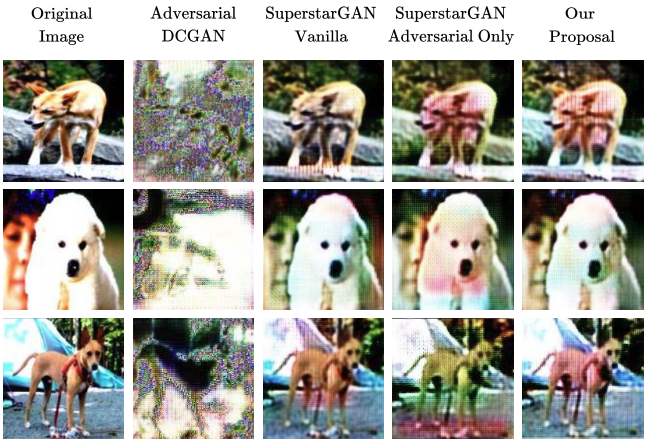

Figure 3: Comparison of images generated by different methods using the SGA attack with a perturbation magnitude of 0.10, targeting the ResNet18 model.

Generating adversarial samples should not rely solely on FR; image quality assessment metrics such as FID and LPIPS are equally crucial. The SuperstarGAN-based images closely mimic the original images, while the *SuperstarGAN Adversarial Only* approach introduces more noticeable perturbations. In contrast, our approach achieves a balanced trade-off—delivering a strong FR while preserving image quality that, although not fully imperceptible, remains visually similar to the original and acceptable to the human eye.

5 Conclusion

This study presents a multi-objective adversarial generative network trained using four well-known adversarial attacks. We evaluated the model's performance against five CNNs using perturbations of varying magnitudes, measuring image quality with LPIPS and FID, and adversarial effectiveness with the FR. Our experimental results demonstrate consistently high FR across all attacks and models, with the best performance nearing 89%. Additionally, LPIPS and FID scores were favourable, reaching approximately 0.23 and 25, respectively. Ablation tests on discriminator weight configurations confirmed the optimal balance between high FR and strong image quality. This balance achieves adversarial traits that are largely imperceptible to the human eye while effectively deceiving models. Furthermore, our results support the theoretical expectation that perturbation magnitude significantly influences evaluation metrics. Future research should explore multi-objective strategies in alternative generative models, such as auto-encoders and VAEs, to assess their potential for improving adversarial image generation and robustness across diverse models.

References

[1] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. Technical report, arXiv, February 2014. arXiv:1312.6199 [cs] type: article.

[2] Xuannan Liu, Yaoyao Zhong, Yuhang Zhang, Lixiong Qin, and Weihong Deng. Enhancing Generalization of Universal Adversarial Perturbation through Gradient Aggregation. Technical report, arXiv, August 2023.

[3] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision, December 2015. arXiv:1512.00567 [cs].

[4] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. Technical report, arXiv, March 2015.

[5] Mohammad Al-Rubaie and J. Morris Chang. Privacy-Preserving Machine Learning: Threats and Solutions. *IEEE Security & Privacy*, 17(2):49–58, March 2019.

[6] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: a simple and accurate method to fool deep neural networks. Technical report, arXiv, July 2016. arXiv:1511.04599 [cs] type: article.

[7] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. arXiv:1706.06083 [stat].

[8] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. A survey on adversarial attacks and defences. *CAAI Transactions on Intelligence Technology*, 6(1):25–45, 2021.

[9] Nicholas Carlini and David Wagner. Towards Evaluating the Robustness of Neural Networks. arXiv:1608.04644 [cs].

[10] Yiran Liu, Xin Feng, Yunlong Wang, Wu Yang, and Di Ming. TRM-UAP: Enhancing the Transferability of Data-Free Universal Adversarial Perturbation via Truncated Ratio Maximization. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4739–4748, October 2023. ISSN: 2380-7504.

[11] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. Technical report, arXiv, March 2017.

[12] Konda Reddy Mopuri, Utsav Garg, and R. Venkatesh Babu. Fast Feature Fool: A data independent approach to universal adversarial perturbations. Technical report, arXiv, July 2017. arXiv:1707.05572 [cs] type: article.

[13] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. Generative Adversarial Networks: An Overview. *IEEE Signal Processing Magazine*, 35(1):53–65, January 2018.

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[15] Soumith Chintala Alec Radford, Luke Metz. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, January 2016. arXiv:1511.06434 [cs].

[16] Kanghyeok Ko, Taesun Yeom, and Minhyeok Lee. Superstargan: Generative adversarial networks for image-to-image translation in large-scale domains. *Neural Networks*, 162:330–339, 2023.

[17] Ugur Demir and Gozde Unal. Patch-based image inpainting with generative adversarial networks, 2018.

[18] Hui Sun, Siman Wu, and Lijun Ma. Adversarial attacks on GAN-based image fusion. *Information Fusion*, 108:102389, August 2024.

[19] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating Adversarial Examples with Adversarial Networks. Technical report, arXiv, February 2019. arXiv:1801.02610 [cs, stat] type: article.

[20] Bingqi Liu, Jiwei Lv, Xinyue Fan, Jie Luo, and Tianyi Zou. Application of an Improved DCGAN for Image Generation. *Mobile Information Systems*, 2022(1):9005552, 2022.

[21] Qingli Yan, Wang Xiong, and Hui-Ming Wang. Secure Indoor Localization Against Adversarial Attacks Using DCGAN. *IEEE Communications Letters*, 29(1):130–134, January 2025.

[22] Biying Deng, Ziyong Ran, Jixin Chen, Desheng Zheng, Qiao Yang, and Lulu Tian. Adversarial examples generation algorithm through dcgan. *Intelligent Automation & Soft Computing*, 30(3), 2021.

[23] José Areia, Leonel Santos, and Rogério Luís de C. Costa. Fooling rate and perceptual similarity: A study on the effectiveness and quality of dcgan-based adversarial attacks. In *Proceedings of the 20th International Conference on Availability, Reliability and Security*, ARES 25, 2025. To be published.

[24] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum, 2018.

[25] Yusuke Tashiro, Yang Song, and Stefano Ermon. Diversity can be transferred: Output diversification for white- and black-box attacks, 2020.

[26] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R. Venkatesh Babu. Nag: Network for adversary generation, 2018.

[27] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie. Generative adversarial perturbations, 2018.

[28] Ishan Durugkar, Ian Gemp, and Sridhar Mahadevan. Generative multi-adversarial networks, 2017.

[29] Isabela Albuquerque, João Monteiro, Thang Doan, Breandan Considine, Tiago Falk, and Ioannis Mitliagkas. Multi-objective training of generative adversarial networks with multiple discriminators, 2019.

[30] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial Training for Free! arXiv:1904.12843 [cs].

[31] Konda Reddy Mopuri, Aditya Ganeshan, and R. Venkatesh Babu. Generalizable Data-free Objective for Crafting Universal Adversarial Perturbations. Technical report, arXiv, July 2018. arXiv:1801.08092 [cs] type: article.

[32] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[33] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric, April 2018. arXiv:1801.03924 [cs].

[34] 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. Technical report, arXiv, January 2015.

[35] FastAI. Imagewoof, 2020.

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[37] Chuan Li and Michael Wand. Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks, April 2016. arXiv:1604.04382 [cs].

[38] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance Normalization: The Missing Ingredient for Fast Stylization, November 2017. arXiv:1607.08022 [cs].

[39] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017. arXiv:1412.6980 [cs].

[40] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1097–1105, 2012.

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)*, 2015.

A Reproducibility

This section outlines the steps required to replicate the our work.

A.1 Code Availability

To support reproducibility and promote further research in this field, we have made the source code publicly available at: https://github.com/ipleiria-ciic/multiobjective-adversarial-gan.

A.2 Dataset Availability

To reproduce our experiments, download the publicly available Imagewoof dataset at: https://github.com/fastai/imagenette?tab=readme-ov-file#imagewoof.

A.3 Installation and Usage

Ensure Python 3.12 is installed along with `pip`. Install the required dependencies using the provided `requirements.txt` file. Then, execute the relevant scripts as follows:

- Use `run_superstargan.sh` to train the adversarial model.
- Use `run_attack.sh` to generate adversarial examples.
- Use `run_encoder.sh` to train the encoder.
- Use `run_testing.sh` to evaluate the images generated.

To modify any parameters during the training process, please refer to the file located at `SupestarGAN/main.py`.