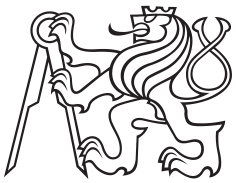
Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Synthetic Image Detection

Bc. Nela Petrželková

Supervisor: Ing. Jan Čech, Ph.D.
January 2024

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Petrželková Nela** Personal ID number: **474506**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Open Informatics**

Specialisation: **Computer Vision and Image Processing**

II. Master’s thesis details

Master’s thesis title in English:

Synthetic Image Detection

Master’s thesis title in Czech:

Detekce syntetických obrázků

Guidelines:

Recently, modern generative models (such as diffusion models or GANs) have provided images of spectacular quality. It is often difficult for a human to distinguish between real images (captured by a camera) and those that are synthesized by a generative model. The goal of the thesis is to propose and implement own classifier or use/modify an existing model to quantitatively evaluate how difficult is to distinguish real and synthesized images. The classification accuracy will be measured under various image degradation conditions (such as blur, compression, noise, subsampling, etc.), and as a function of a size of the image, either isolated patches or blended into a real image. Optionally, experiment with adversarial attacks. It is recommended to narrow down the domain of images, for example, to faces.

Bibliography / sources:

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In Proc. CVPR, 2022.
[2] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, Taesung Park. Scaling up GANs for Text-to-Image Synthesis. In Proc. CVPR, 2023.
[3] Xiao Guo, Xiaohong Liu, Zhiyuan Ren, Steven Grosz, Iacopo Masi, Xiaoming Liu. Hierarchical Fine-Grained Image Forgery Detection and Localization. In Proc. CVPR, 2023.
[4] Sheng-Yu Wang, Oliver Wang, Richard Zhang, Andrew Owens, Alexei A. Efros. CNN-generated images are surprisingly easy to spot...for now. In Proc. CVPR, 2020.

Name and workplace of master’s thesis supervisor:

Ing. Jan Čech, Ph.D. Visual Recognition Group FEE

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **15.09.2023** Deadline for master's thesis submission: **09.01.2024**

Assignment valid until: **16.02.2025**

Ing. Jan Čech, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Acknowledgements

I would like to express my sincerest gratitude to my supervisor, Ing. Jan Čech, Ph.D., for the guidance, time, and support he has given me in the last few months. I would also like to thank my friends and family for supporting me throughout my studies.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instruction for observing the ethical principles in the preparation of university theses.

This includes leveraging tools like OpenAI's ChatGPT for linguistic refinement.

Prague, January 9, 2024

Prohlašuji, že jsem předloženou práci vypracovala samostatně a že jsem uvedla veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Toto zahrnuje použití ChatGPT od OpenAI pro jazykovou úpravu textu.

V Praze, 9. ledna 2024

.....

Abstract

This thesis presents several experiments on detecting synthetic face images. We find that a simple classification model of the standard ResNET-50 architecture trained on a specific image generator can achieve near-perfect accuracy in separating synthetic and real images. The model also handles common image distortions (reduced resolution, compression) by using data augmentation. Moreover, partial manipulations, where synthetic images are blended into the real ones, are identified and the area of the manipulation is localized by a simple model of standard YOLO architecture.

However, we also find that the model is vulnerable to adversarial attacks and does not generalize to data from unseen generators. Failure to generalize to detect images produced by a newer generator also occurs for recent state-of-the-art methods, which we tested on Realistic Vision, a fine-tuned version of Stable Diffusion image generator.

Keywords: generative modelling, diffusion models, adversarial attacks, deepfakes, detection, localization, synthetic images

Supervisor: Ing. Jan Čech, Ph.D.

Abstrakt

Tato práce prezentuje několik experimentů týkajících se detekce syntetických obrázků obličejů. Bylo zjištěno, že klasifikační model standardní architektury ResNET-50, trénovaný na specifickém generátoru obrázků, může dosáhnout téměř dokonalé přesnosti při rozpoznávání syntetických a skutečných obrázků. Model také zvládá běžné deformace obrazu (snížené rozlišení, kompresi) pomocí augmentace dat. Kromě toho jsou částečné manipulace, kdy jsou syntetické obrázky smíchány se skutečnými, identifikovány a oblast manipulace je lokalizována pomocí jednoduchého modelu standardní architektury YOLO.

Nicméně bylo zjištěno, že model je zranitelný vůči adversariálním útokům a nefunguje dobře pro generátory, jejichž obrázky nebyly zahrnuty v trénovací sadě. Nedostatek schopnosti generalizace pro detekci obrázků vytvořených novějším generátorem se vyskytuje i u nejnovějších metod, které jsme testovali na Realistic Vision, vylepšené verzi generátoru obrázků Stable Diffusion.

Klíčová slova: generativní modelování, difúzní modely, adversariální útoky, deepfakes, detekce, lokalizace, syntetické obrázky

Překlad názvu: Detekce syntetických obrázků

Contents

Chapter 1

Introduction

Image synthesis has made significant progress in recent years, thanks to the advances of generative models such as Generative Adversarial Networks (GANs) [33] and Diffusion Models [51]. Synthesized images are becoming increasingly realistic and hardly distinguishable from real ones by an average human and even by an expert with naked eyes. The generation of photorealistic images of people through artificial intelligence poses a dual-edged sword. On one hand, it showcases the impressive strides we have made in technology, offering tools for creativity and innovation across various fields. Creating synthetic data reduces the overall costs of acquiring data, and it is scalable and accurate. One can generate specific corner cases that are otherwise hard to collect. Some relevant application areas of generative models are, for instance:

- **Training and enhancing AI models.** One of the primary challenges in training effective machine learning models is obtaining a sufficiently large and diverse dataset [62]. This can be expensive and time-consuming. Synthetic data generation can fill this gap by creating a wide variety of data instances that might not be readily available or easy to collect, and it can be useful to create samples for underrepresented classes to avoid imbalanced and biased datasets. For instance, in autonomous vehicle technology, it is crucial to expose the AI to a vast range of driving conditions, scenarios, and environments. Creating these scenarios in the real world would be impractical and dangerous, but synthetic data can simulate rare or hazardous conditions such as extreme weather, lighting variations, or unexpected pedestrian behavior without any risk [37].
- **Entertainment and media production.** In the film and gaming industries, synthetic imagery can create realistic characters or environments, reducing production costs and time. It is also used to create special effects and animations. For example, it can help with dubbing movies by altering lip movement to match translated dialogues [67]. In the area of virtual reality, synthetic imagery is crucial for creating immersive and interactive experiences.
- **Educational purposes.** Synthesizing historical footage, images, or figures can be used in documentaries or educational materials to recre-

ate scenes or events that were never recorded or have been lost over time [5]. Other scenarios include creating realistic training environments for procedures ranging from surgical practices [53] to military exercises, providing a safe and controlled learning environment.

- **Fashion industry.** For virtual fashion trials, AI can generate images of models wearing different outfits, helping customers visualize products without the need for physical prototypes [29].

- **Medical applications.** One concern related to data collection can be privacy, for example, in fields such as healthcare. Synthetic data allows the training of AI models without compromising patient confidentiality. By generating realistic but artificial medical images or patient records, researchers and developers can work on sensitive applications such as diagnostic tools or treatment planning systems while adhering to privacy regulations [9].

On the other hand, this progress also poses serious threats to individuals and society. Synthesized images or videos, a.k.a. 'deep fakes' [45] can be used for malicious purposes and can lead to potentially undermining public trust in media, invading privacy, or perpetuating misinformation that alters public opinion or affects international relations. Some cases of misuse of synthetic imagery or videos are fake porn [39], fake video calls [6, 61], fake news [66] (see Fig. 1.1 for an example) or fake videos in election campaigns [49, 42]. Therefore, it is important to develop effective and robust methods to detect and expose fake images to prevent potential malicious use, especially in the domain of faces, which are often the target of attacks.

In this thesis, we present an experimental study in which we show some key properties of neural fake face detectors. We use a standard classifier, and we do not primarily aim to push accuracy on standard datasets, but rather focus on other aspects of the problem. In particular, we investigate the generalization ability of the detector to unseen generators, the accuracy with respect to different image degradations and input sizes, the vulnerability to adversarial attacks, and the localization of fake regions in the case of partial manipulation of real images.

In addition to detectors that are accurate in spotting images of recent generators (namely, the Stable diffusion [51] — Realistic Vision [12] checkpoint), our main contribution is a thorough analysis of forgery detectors that revealed many intriguing properties. The contributions are summarized below.

1. **Cross-generator-detector testing.** We show that while it is surprisingly easy to train a detector for a specific synthetic image generator, its accuracy drops dramatically when tested on images produced by a different generator, which was not trained for. This effect is partially reduced by training the detector on images generated by multiple different generators. Then the detector generalizes better to unseen generated images. We quantify this effect and show learning curves that demonstrate the accuracy as a function of a number of training images.

Figure 1.1: The deepfake with president Zelensky. It appeared on the hacked website of Ukrainian TV network Ukrayina 24 in March 2022. In the video the fake Zelensky called Ukrainians to put down their weapons during an ongoing military conflict. Image source: [72].

2. **Robustness to input size and degradations (blur, compression) analysis.** We tested the detector against blur, JPEG compression, and shrinking the input patch size by downsizing or masking the input image. The detector is surprisingly robust to the degradations and is even more robust when the degradations are added to the training set as data augmentation. For instance, our detector is able to spot synthetic images from a patch as small as 25×25 px with an accuracy of about 70%.

3. **Adversarial attacks vulnerability checking.** We demonstrate that adversarial images can be easily found and are able to fool the detector to classify a synthetic image as a real one. Moreover, we show that residuals found for a particular fake image have an adversarial effect on other images and can also fool a different model of a very different architecture. We tested for convolutional networks and a vision transformer. We also examine some defense mechanisms against these types of attacks.

4. **Localizing partial manipulations.** A likely scenario of a fraudulent act is to blend synthetic images into real photos. Therefore, we prepared a set of partially manipulated images using state-of-the-art inpainting models replacing key regions of the face (eyes, nose, mouth, etc.). We show that such images are easily spotted despite the manipulated area being small. Moreover, the manipulated area is localized within the image with high accuracy. Furthermore, we measure how varying the size of the manipulated areas affects the model accuracy.

The remainder of the thesis is structured as follows. Chapter 2 focuses on the related work and current state of the art in the field of synthetic images and their detection. Chapter 3 gives the necessary technical background. In Chapter 4, we experimentally analyze properties of synthetic image detectors, adversarial attacks, and localizing fake parts of images. The thesis is concluded in Chapter 5.

Chapter 2

Related Work

In conjunction with the rapid development of high-quality synthetic image generators, research on the detection of fake images has become very active. For a comprehensive review, we refer to recent surveys [74, 1, 41]. In this section, we review some of the existing methods and challenges for this problem.

Historically, before the boom of deep learning, fake image detection focused on detecting "doctored images" that were manually edited or manipulated from images captured by cameras. These methods relied on various clues, such as steganographic features, compression artifacts, or inconsistencies in lighting or shadows [20, 50]. However, these methods are not effective against synthetic images that are generated from scratch or with minimal human intervention.

Forensic low-level signal detectors are another class of methods that exploit the spectral signatures of synthetic images. Inspiration probably came from the forensic recognition of a camera device [8]. More recently, researchers discovered that the residual spectra of synthetic images contain typical anomalies, which creates a spectral fingerprint of a synthetic generator [68, 14, 13]. Examples of some of the spectra are shown in Fig. 2.1. A simple method based on the spectral fingerprint [17] reports a high detection accuracy. However, these methods may not be robust to image transformations, such as resizing or compression, that can alter the signal characteristics.

In a similar spirit, other more recent methods suggested that the information for fake image detection is deeply embedded in the image signals and can be detected independently of the image context. For example, work [7] proposed a method that uses a convolutional neural network with a narrow receptive field to detect fake images from signal patches and showed that some regions, such as hair, are more discriminative than others.

Another paper [68] has shown that the detection of synthetic images generated by generative adversarial networks (GANs) is relatively easy, as they exhibit certain distinctive features and a standard CNN classifier can be easily trained to capture this difference. The paper even shows a good generalization ability, in case the model is tested on images produced by a generator for which it was not trained for. However, they used only GAN-based generators. In this work, we will show that the generalization of

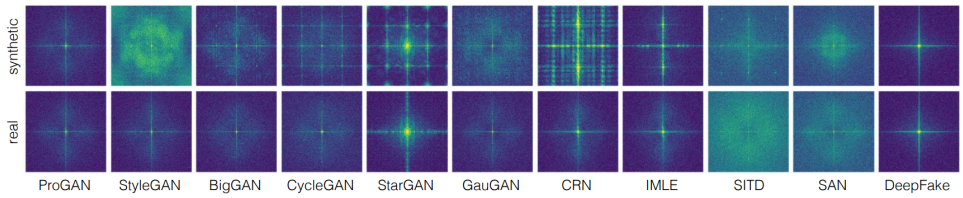


Figure 2.1: The average spectra of each high-pass filtered image. For both the real and fake images. Image source: [68].

detectors to unseen generators between recent models (especially of different architectures GANs/diffusion models) is poor. We show that recent state-of-the-art synthetic image detectors fail completely when tested on images produced by novel unseen generators.

Besides detection, some recent works have also addressed the localization of fake images, which aims to segment the manipulated areas of the real images. The problem is challenging considering possibly a small area of manipulation. Some methods do not use any special architecture for localization but rely on post-processing techniques. Recent paper [60] compares a popular Grad-CAM [54] to highlight the regions that contribute to the classification decision and the scanning technique of [7] to localize synthetic regions in partially manipulated images. Other methods use more complex architectures, such as multi-branch network [23], or dense self-attention network [25], to explicitly learn the localization maps. Paper [38] fine-tunes a large segmentation model (SAM) [36] to adapt it to the fake image domain. We show that precise localization results are achieved for relatively small regions using a simple YOLO-based architecture [48], namely YOLOv8 [64], as long as the fake images are composed of images produced by the same generator model that we trained on.

Deep neural networks are known to be vulnerable to adversarial attacks [59], which are small imperceptible perturbations of the input that cause a network to make a wrong prediction. This problem has been extensively studied in various domains, such as image classification [22], object detection [70], or face recognition [16]. In this work, we show that this vulnerability also applies to the fake image detection domain and that a common way of generating adversarial examples can fool the detectors into classifying fake images as real.

For other thoughts on the social impact of deepfakes, we refer to e.g., [24, 34].

Chapter 3

Technical Background

In this chapter, we will introduce key concepts that we worked with during our experiments. Firstly, we will have a closer look at synthetic image generation and its two prominent techniques in recent years – generative adversarial networks and diffusion models. Then, we will describe adversarial attacks – in particular, the fast gradient sign method (FGSM) and Limited-memory BFGS (L-BFGS).

3.1 Generating Synthetic Images

In recent years, the field of synthetic image generation has seen significant advancements, driven by the development of sophisticated algorithms and deep learning techniques. At the core of these developments are two important methods: generative adversarial networks and diffusion models. In this section, we take a closer look at how these methods work internally.

3.1.1 Generative Adversarial Networks (GANs)

Over the past few years, Generative Adversarial Networks (GANs), first introduced by Goodfellow et al. [21], have become increasingly popular in various domains such as image processing, language understanding, semantic segmentation, or creating synthetic time series data [69]. Particularly, for images, GANs have been highly effective, applied in tasks such as image synthesis [31, 75], image-to-image translation [30, 76], face image completion [73], and image super-resolution [40].

A typical GAN setup includes two modules: a *discriminator* and a *generator*. Both are neural networks that compete against each other, as illustrated in Fig. 3.1. The generator's goal is to create images from random noise that look so real, they can fool the discriminator. Meanwhile, the discriminator acts as a binary classifier, trying to distinguish real images and those made by the generator.

The generator G is designed to convert input noise vectors $\mathbf{z} \in \mathcal{Z}$, drawn from a distribution $p_{\mathbf{z}}$ (commonly Gaussian), into synthetic images $\mathbf{s} \in \mathcal{S}$. The

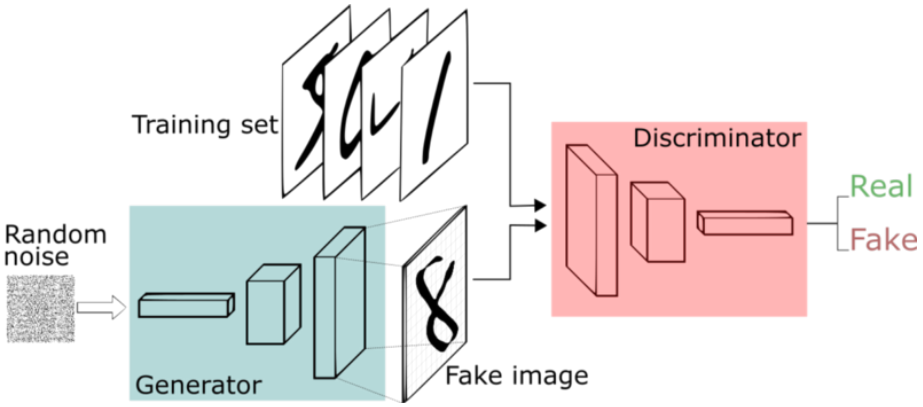


Figure 3.1: A typical architecture of a generative adversarial network. In the blue box is the generator G that gets a random vector as an input and generates a synthetic image. The task of the discriminator D, given an image as an input, is to distinguish whether it is a real or a synthesized image. The figure is adopted from [55].

generator is parameterized by θ_g, which defines the probability distribution $p_{\mathbf{s}}$ of the generated samples $\mathbf{s} \in \mathcal{S}$. Formally, this transformation by the generator is defined as

$$G(\mathbf{z}; \theta_g) : \mathcal{Z} \to \mathcal{S}. \tag{3.1}$$

The discriminator D is a binary classifier, parametrized by θ_d and its task is to distinguish between real and generated images,

$$D(\mathbf{x}; \theta_d) : \{\mathcal{X}, \mathcal{S}\} \to [0; 1]. \tag{3.2}$$

where $\mathbf{x}$ is an image and $D(\mathbf{x})$ predicts the probability that $\mathbf{x}$ came from real data distribution $p_{\mathbf{x}}$ rather than synthetic distribution $p_{\mathbf{s}}$.

The objective of a GAN is to learn the generator distribution $p_{\mathbf{s}}$ to be as close as possible to the distribution $p_{\mathbf{x}}$ of real data, i.e. to transform the latent vectors to a manifold that resembles the real data in the best way possible. This way the generator G would fool the discriminator D by minimizing $\log(1 - D(G(\mathbf{z})))$. At the same time, we want the decisions of the discriminator D to be correct by maximizing $\log(1 - D(G(\mathbf{z})))$. From that follows that D and G play the following two-player minimax game with joint loss function $V(G, D)$:

$$\min_G \max_D V(G, D) = \mathbb{E}_{\mathbf{x} \sim p_{\mathbf{x}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}}[1 - \log D(G(\mathbf{z}))]. \tag{3.3}$$

Despite their simplicity, training GANs can be challenging. Because G and D update their parameters θ_g and θ_d independently of each other, updating the gradient of both models at the same time does not guarantee convergence. Another problem is the vanishing gradient that can appear particularly at

the beginning of the training. Before G learns some distribution resembling of the real data distribution, it generates random images, making decisions of D easy. If the discriminator D always classifies the image correctly, the loss function will have a value of zero and the gradient will be zero-valued too. The next challenge in training GANs is avoiding *mode collapse*. Mode collapse is a phenomenon that happens when the generator is able to trick the discriminator, but it generates images with low diversity and is not able to represent the real data distribution.

To solve these issues, various techniques like feature matching, minibatch discrimination, and historical averaging have been introduced. *Wasserstein GANs* [3] mitigated the problem of mode collapse and improved the learning stability of the model.

Conditional GANs. Conditional GANs (cGANs) [43] enhance the capabilities of the traditional GAN framework by introducing conditions into the image generation process. These conditions, often in the form of class labels $\mathbf{y}$, allow for more controlled and targeted image synthesis. This is particularly useful in applications where the desired output is specific to certain attributes or categories.

The objective function of a cGAN is modified to incorporate these conditions, which can be formally expressed as:

$$\min_G \max_D V(G,D) = \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[\log D(\mathbf{x}|\mathbf{y})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[1 - \log D(G(\mathbf{z}|\mathbf{y})|\mathbf{y})]. \quad (3.4)$$

In this equation, $D(\mathbf{x}|\mathbf{y})$ represents the discriminator's estimation of the probability that $\mathbf{x}$ is a real image, given the condition $\mathbf{y}$, and $G(\mathbf{z}|\mathbf{y})$ denotes the generator's output given the noise vector $\mathbf{z}$ and condition $\mathbf{y}$. The incorporation of $\mathbf{y}$ allows the GAN to generate images that are not only realistic but also tailored to specific classes or characteristics dictated by $\mathbf{y}$.

Such a framework expands the utility of GANs in various fields, enabling the generation of context-specific images, which is crucial in scenarios like personalized content creation, targeted data augmentation for machine learning models, and domain-specific image synthesis.

PG-GAN

In 2018, Karras et al. [31] introduced progressively growing GANs (PG-GAN). The main idea behind progressively growing GANs is that initially both the generator G and the discriminator D have a low spatial resolution of 4×4 pixels, and during training, the convolutional layers are incrementally added to G and D, while all existing layers remain trainable. The gradual addition of layers leads to increasing the spatial resolution of the generated images. It also enables the model to efficiently acquire general information at a high level and subsequently acquire more specific details. Another advantage is the reduced training time. This process is visualized in Fig. 3.2.

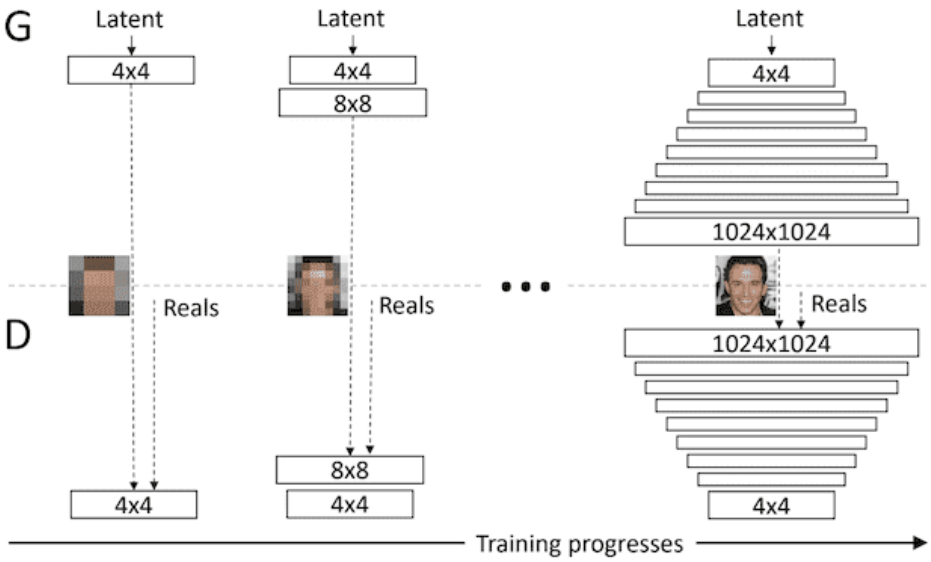


Figure 3.2: Training of PG-GAN. At the beginning both G and D have low spatial resolution of 4×4 pixels. During training additional layers are incrementally added, increasing the spatial resolution of the generated images.

StyleGAN

In March 2019, Karras et al. introduced StyleGAN [32], a novel generator architecture, that was a significant advancement in the field of data-driven unconditional generative image modeling. The new architecture enables one to automatically learn a separation between high-level features such as pose or identity when trained on human faces and stochastic variation such as hair structure or freckles, which allows for meaningful interpolation operations and scale-specific mixing in the image space [46]. It has been observed that when linearly interpolating two latent vectors $\mathbf{z_1}$ and $\mathbf{z_2}$,

$$\mathbf{z} = \alpha \mathbf{z_1} + (1 - \alpha)\mathbf{z_2}, \quad 0 \leq \alpha \leq 1, \tag{3.5}$$

the corresponding generated images $G(\mathbf{z})$ change continuously. That means that the semantics contained in the image also change gradually [32]. An example of this property is shown in Fig. 3.3.

While the objective function and the discriminator of StyleGAN remain consistent with typical unconditional GANs, the StyleGAN generator differs significantly from a traditional generator in its architecture, as shown in Fig. 3.4. It is composed of a mapping network f and a synthesis network g. It first transforms a latent code $\mathbf{z} \in \mathcal{Z}$ into $\mathbf{w} \in \mathcal{W}$ using the mapping network $f : \mathcal{Z} \to \mathcal{W}$, where both vectors are 512-dimensional. The advantage of this approach is that $\mathcal{W}$, unlike $\mathcal{Z}$, is not constrained by any specific probability distribution, which helps in the separation of various features.

Figure 3.3: Interpolating between latent codes od StyleGAN. The left- and right-most images as randomly generated from StyleGAN. Linear interpolation between their generating latent code changes the images smoothly in the image space as well.

Learned affine transformations then transform $\mathbf{w}$ to styles that control adaptive instance normalization (AdaIN) [28]. To add detailed stochastic features to the images, noise vectors are incorporated at each convolutional layer's output.

Through this architecture, specific styles can be adjusted at different scales to manipulate the image synthesis process. The combination of the mapping network and affine transformations effectively samples each style from a learned distribution, and the style-based architecture enables to generate an image composed from those styles.

3.1.2 Diffusion Models

Diffusion models have emerged as a significant breakthrough in the field of generative models, especially for tasks involving image synthesis. The concept of utilizing diffusion models for image generation first emerged in the literature in 2015 [56], however, five years later, these models experienced a significant surge in popularity and recognition in the field thanks to Ho et al. [27] and the introduction of denoising diffusion probabilistic models (DDPMs). These models represent a distinct approach from traditional generative methods such as GANs. Diffusion models are inspired by the physical process of diffusion, and they have shown remarkable results in generating high-quality, high-fidelity images.

The core principle of diffusion models involves a gradual, stepwise process that transforms a simple distribution (like Gaussian noise) into a complex data distribution (such as natural images). This transformation is achieved through a series of forward and reverse diffusion steps.

In the forward process, the model incrementally adds noise to the data, gradually transforming it into a Gaussian distribution. Conversely, the reverse process involves a neural network learning to reverse these diffusion steps, effectively reconstructing the original data from the noise.

Mathematically, the forward diffusion process can be described as a Markov chain that incrementally adds Gaussian noise to the data at each step. This process is visualized in Fig. 3.5. Given a data point sampled from a real data distribution $\mathbf{x}_0 \sim q(\mathbf{x})$, we define a forward diffusion process in which we add a small amount of Gaussian noise to the sample in T steps, producing a sequence of noisy samples $\mathbf{x}_1, \ldots, \mathbf{x}_T$. The step sizes are controlled by a

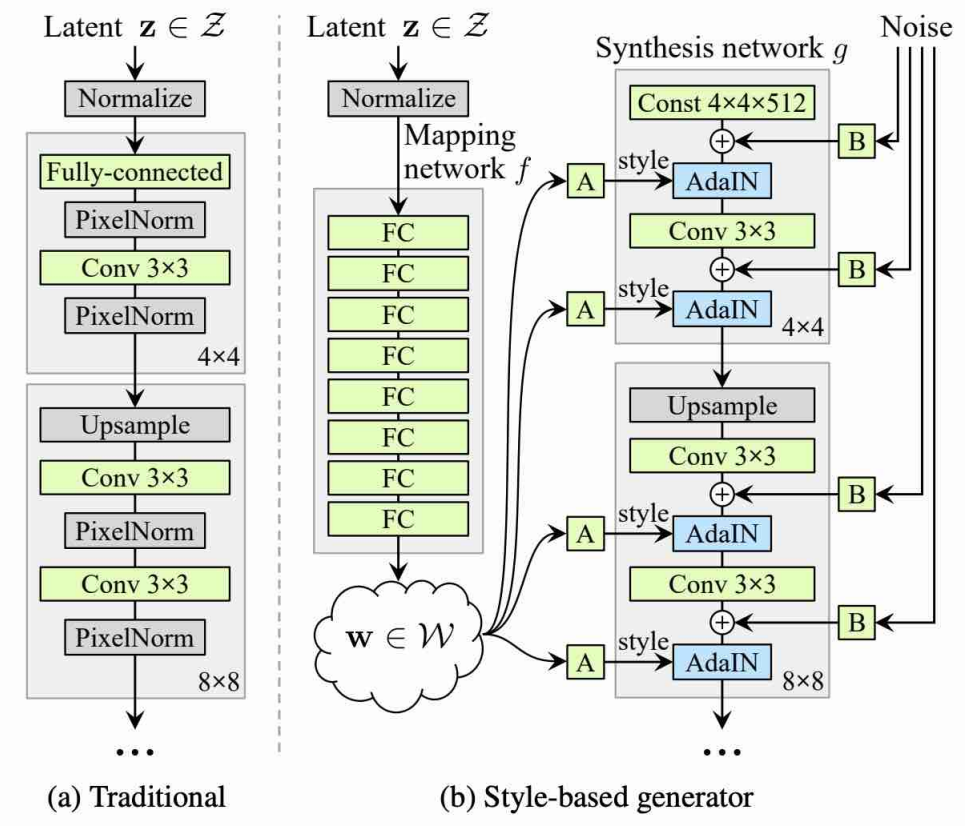


Figure 3.4: Comparison of the traditional generator with a StyleGAN generator. In the traditional generator, the latent code $\mathbf{z}$ is passed directly to the input layer, to the style-based generator. In the style-based generator, the latent vector is first mapped to intermediate latent code $\mathbf{w} \in \mathcal{W}$. An affine transformation A is applied to $\mathbf{w}$ and the result is then forwarded to a convolutional layer with AdaIN. This is done for each convolutional layer separately. Latent code $\mathbf{w}$ is the same, but the affine transformations are different for each convolutional layer. After each convolution and before non-linearity, Gaussian noise is injected to provide additional diversity of the results [46]. The figure is adopted from [32].

variance schedule $\{\beta_t \in (0,1)\}_{t=1}^{T}$. Then, the forward process is

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}), \tag{3.6}$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}). \tag{3.7}$$

The reverse process, which is of primary interest, is where the model learns to generate data. It involves training a neural network to predict the noise that was added at each step of the forward process, thereby allowing it to reverse these steps and generate samples from the target distribution. This is visualized in Fig. 3.6.

If we can reverse the forward process mentioned previously, and sample from $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$, we will be able to recreate the true sample from a Gaussian noise

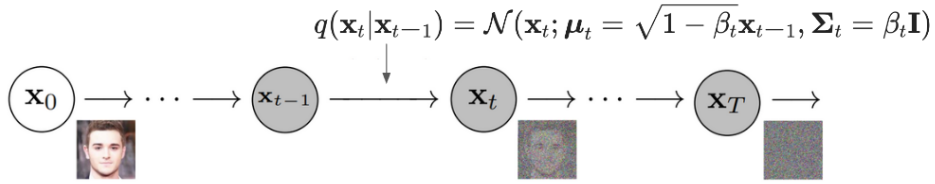


Figure 3.5: The Markov chain of a forward diffusion process. The original image gradually loses its distinguishable features by slowly adding noise in multiple steps. Image source: [71]

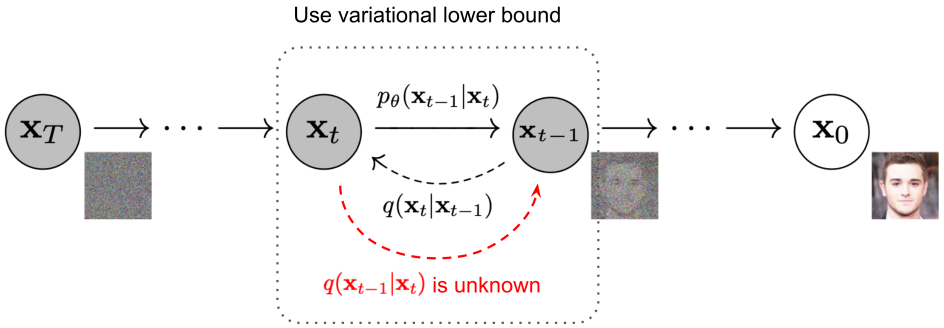


Figure 3.6: Visualization of a reverse diffusion process. The goal is to recreate the true sample from a noisy input in multiple steps. Image source: [71]

input, $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Unfortunately, we cannot easily estimate $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ and therefore we need to learn a model p_θ to approximate these conditional probabilities in order to run the reverse diffusion process

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t), \tag{3.8}$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)). \tag{3.9}$$

However, the process of generating images with this specific approach takes a long time compared to e.g., GANs. As Song et al. 2020 [58] wrote: "For example, it takes around 20 hours to sample 50k images of size 32 × 32 from a DDPM, but less than a minute to do so from a GAN on an Nvidia 2080 Ti GPU." That is when *denoising diffusion implicit model (DDIM)* [58] comes into play. DDIMs enable to sample only a subset of the diffusion steps during the generation process, making the inference much faster.

Latent Diffusion Models and Stable Diffusion

The Latent Diffusion Model (LDM) [51] builds on DDIM and also implements the diffusion process in a latent space, rather than a pixel space, as shown in Fig. 3.7. This approach results in more efficient training and faster inference. The rationale behind a latent diffusion model is motivated by the

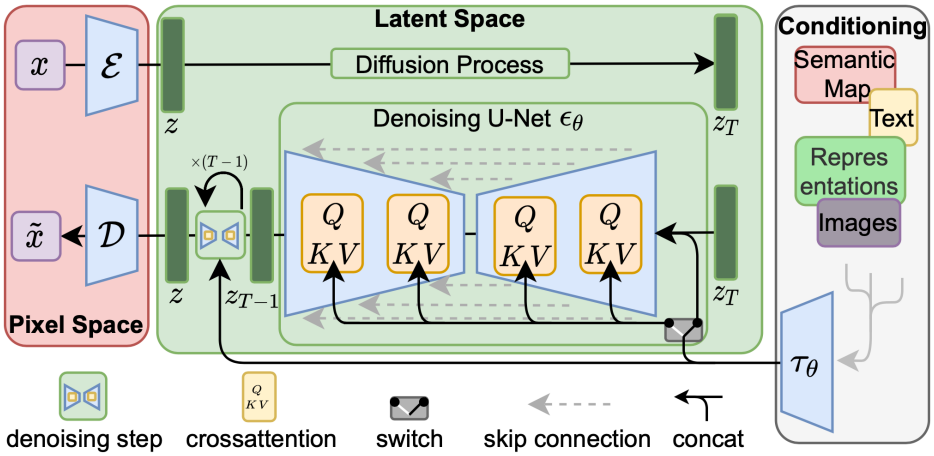


Figure 3.7: The architecture of a latent diffusion model. An embedding of a conditioning input is created with an encoder τ_θ. The embedding is then injected to intermediate layers of the denoising U-Net ϵ_θ with the cross-attention mechanism. The forward and backward diffusion processes are carried out in the latent space $\mathbf{z}$. Image taken from [51].

understanding that while a significant portion of the image data contributes to fine-grained perceptual details, its core semantic and conceptual structure still remains after an aggressive compression. LDMs thus effectively separate perceptual and semantic features in the image generation. This is achieved by initially reducing pixel-level details using an autoencoder. Subsequently, it applies the diffusion process within the resulting latent space to synthesize or modify the semantic content of the image. This two-step approach, focusing first on removing redundancy at the pixel level and then on handling semantic aspects through diffusion, offers a more resource-efficient way to generate high-quality images. This was an important breakthrough because it enabled running of these models on consumer hardware as they are less computationally demanding than the models that operate in the pixel space.

As stated previously, the main difference compared to vanilla diffusion models is that the encoder E compresses the high-dimensional input image $\mathbf{x} \in \mathbb{R}^{H \times W \times 3}$ into a reduced 2D latent space $\mathbf{z} = \mathcal{E}(\mathbf{x}) \in \mathbb{R}^{h \times w \times c}$, where the downscaling factor $f = H/h = W/w = 2^m, m \in \mathbb{N}$. The decoder D then aims to reconstruct the image from this latent space representation, $\hat{\mathbf{x}} = D(\mathbf{z})$.

The diffusion process is conducted on the latent space vector $\mathbf{z}$. A temporally conditioned U-Net, which incorporates a cross-attention mechanism, enables flexible and dynamic conditioning for diverse image generation tasks. This is particularly useful for encoding various types of conditioning information, such as class labels, text, layouts, or semantic maps.

Each type of conditioning information is associated with a domain-specific encoder τ_θ which projects the conditioning input $\mathbf{y}$ onto an intermediate representation $\tau_\theta(\mathbf{y}) \in \mathbb{R}^{M \times d_\tau}$. This representation is then mapped to the intermediate layers of the U-Net via a cross-attention layer.

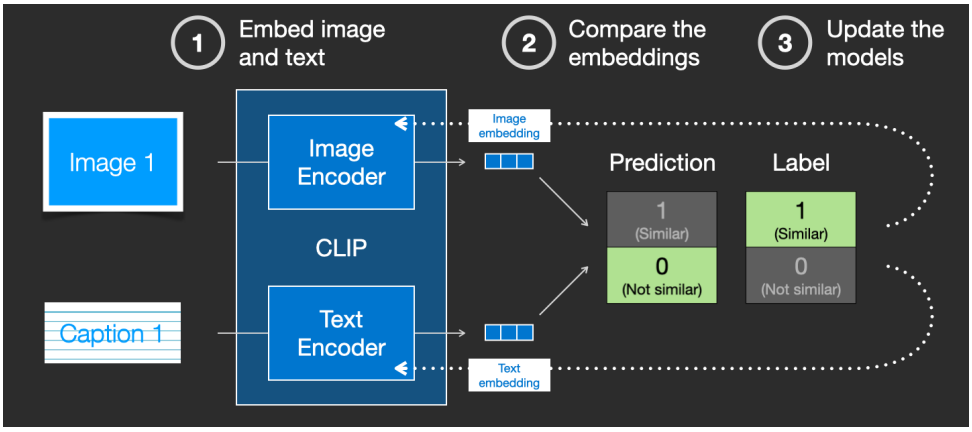


Figure 3.8: Training of CLIP. During training, CLIP learns to embed an image and its corresponding caption. The goal is that the image embedding and text embedding have high similarity scores. Image source: [2].

Probably the most prominent conditioning is text. CLIP [47] tokenizer is used to create the intermediate representation $\tau_\theta(\mathbf{y})$ of the text prompt. CLIP is trained on a dataset of 400M images and their captions that were obtained by crawling the web along with their "alt" tags. The key idea of CLIP is that it is a combination of an image encoder and a text encoder. The aim is to make the embeddings of the image with the corresponding caption similar to each other. During training, the embeddings are compared, and their similarity score is measured. This process is visualized in Fig. 3.8. Stable Diffusion then uses the CLIP text encoder to obtain the tokenized representation of the text prompt. The representation is limited to 77 tokens where one token is a 768-dimensional vector. These representations are injected into the intermediate layers of U-Net by the cross-attention mechanism [65].

There are many Stable Diffusion checkpoints (or models) available, and every day new ones are appearing. The difference between them is the data on which they were trained. In 2023, Stable Diffusion v2 was introduced. It offers synthesis of higher resolution data and it uses OpenClip [10] text tokenizer instead of CLIP. OpenClip was trained on publicly available dataset and NSFW data samples were filtered out from the dataset. Stable Diffusion community found, that some images look worse in the V2 model and it is harder to generate higher quality results for keywords like celebrity names, and in general, it is easier to synthesize images in artistic style. [44] In our work, we use two checkpoints from Stable Diffusion V1. First is Stable Diffusion V1.4, which is considered to be the first publicly available Stable Diffusion model. The second one is Realistic Vision 5.1, which is a model based on Stable Diffusion V1.5 and focuses on generating photo-realistic images.

3.2 Adversarial Attacks

Adversarial attacks in image classification represent a sophisticated form of interference where images are subtly modified to mislead machine learning models into incorrect predictions. These modifications are usually imperceptible to humans but can cause a model to incorrectly label data with high confidence. This is visualized in Fig. 3.9.

In the context of distinguishing real images from artificially generated images of people, such attacks can have significant implications. They might, for instance, trick a system into accepting a fake image as real, raising concerns in areas like digital security and content authentication. As the field advances, one promising area is the use of artificial intelligence not just to create but also to automatically detect and respond to adversarial attacks. This approach could lead to more dynamic and robust defense mechanisms, adapting in real time to the ever-evolving landscape of adversarial threats in image classification.

The challenge of defending against these attacks is ongoing. One common strategy is to strengthen the model's resilience by training it with a variety of adversarial examples, a process known as adversarial training. Other defensive measures include introducing an image transformation on inference (e.g. Gaussian blur) in order to mitigate the adversarial pattern. We will study these types of defenses more in Section 4.4.

As mentioned previously, creating adversarial samples typically involves subtly tweaking the image data. The methods that are often used are simple but effective Fast Gradient Sign Method (FGSM) or more elaborate iterative techniques. These methods adjust the image in minimal ways to alter the classification outcome without the changes being noticeable to human viewers. Some particular techniques are for example Jacobian-based Saliency Map Attack (JSMA), Deepfool, or L-BFGS. We will elaborate more on the L-BFGS later in this section.

3.2.1 Fast Gradient Sign Method (FGSM)

The Fast Gradient Sign Method (FGSM) is a simple, yet effective method of creating adversarial samples. It was first introduced by Goodfellow et al. [22]. The core idea behind this method is to exploit the gradients of the neural network to slightly alter the input data in a way that it leads to incorrect output from the network. The authors recognized that neural networks are linear models to a large extent despite the nonlinear activation functions, meaning that overall behavior, especially in deep networks, can be approximated by a linear model. This linearity makes them susceptible to adversarial perturbations [22]. Moreover, they observed that in high-dimensional spaces, even small perturbations can lead to significant changes in the output. This is due to the cumulative effect of small changes across many dimensions, pixels, in our case.

The objective was to find an input change that would maximally increase

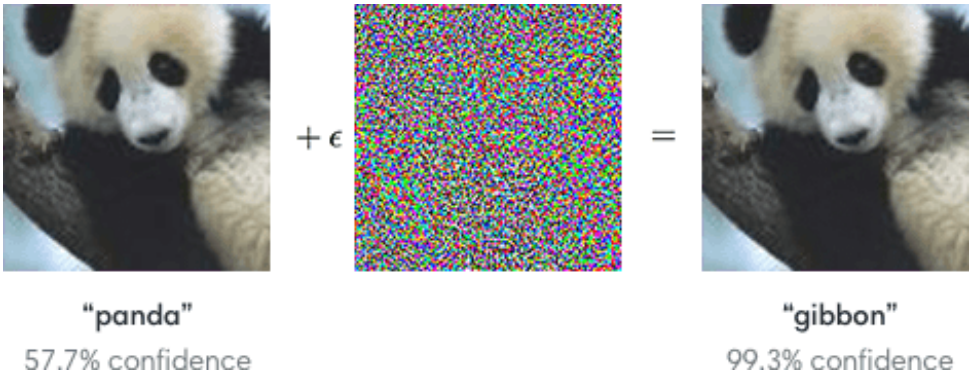


Figure 3.9: An example of an adversarial attack. We take an image of a panda, that was correctly classified by the model with 57.7% confidence. By adding the constructed adversarial residuum, we change the model decision to "gibbon" with 99.3% confidence. For the human eye, the difference in the image is barely noticeable. Image taken from [22].

the loss of the model, leading to a higher chance of misclassification. The gradient of the loss function with respect to the input image indicates the direction in which a small change would increase the loss. To keep the method efficient, they proposed to use the sign of the gradient, which provides a good approximation and is computationally simple. This whole term is multiplied by a small constant $\epsilon \in (0; 1)$ that determines the strength of the adversarial attack. By tuning this value, one can ensure that the perturbation is small enough to keep the adversarial image visually similar to the original sample $\mathbf{x}$, yet sufficiently large to fool the model into the wrong classification. Note, that the input image $\mathbf{x}$ is normalized to values between 0 and 1. Then the adversarial sample is obtained as

$$\mathbf{x}' = \mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y)), \tag{3.10}$$

where θ represents the model parameters and y the target class.

FGSM is a straightforward and computationally efficient method for generating adversarial examples and requires only a single step to compute the perturbation. This makes it a widely used technique for testing and enhancing the robustness of neural networks against adversarial attacks.

3.2.2 Limited-memory BFGS (L-BFGS)

L-BFGS (Limited-memory Broyden–Fletcher–Goldfarb–Shanno) is a well-known optimization algorithm in machine learning, particularly suited for problems with large numbers of variables, such as high-dimensional image data. Szegedy et al. [59] utilized this algorithm to optimize the perturbation added to the input image. Precisely, they use box-constrained L-BFGS, which refers to the constraint on pixel values, typically from 0 to 1 for normalized images.

The core idea behind this approach is to find the smallest possible perturbation (also referred to as residuum) $\mathbf{r}$ of an original image $\mathbf{x}$ that leads to

misclassification as a specific target class l. This involves minimizing a loss function that represents the distance between the original and adversarial images, subject to the condition that the adversarial image is misclassified by the classification model $f : \mathbb{R}^m \to \{1, \ldots, k\}$, where m is the number of pixels in the image, k is the number of classes. The optimization problem can be expressed as

$$\min_{\mathbf{r}} \quad ||\mathbf{r}||_2 \tag{3.11}$$

$$\text{subject to:} \quad \mathbf{x} + \mathbf{r} \in [0, 1]^m \tag{3.12}$$

$$f(\mathbf{x} + \mathbf{r}) = l. \tag{3.13}$$

However, solving this optimization problem directly can be computationally intensive, especially for high-dimensional data like images, and this is where L-BFGS comes into play. It is used to iteratively approximate the perturbation $\mathbf{r}$ that will lead to misclassification, changing the objective to

$$\min_{\mathbf{r}} \quad c||\mathbf{r}||_2 + \text{loss}_f(\mathbf{x} + \mathbf{r}, l) \tag{3.14}$$

$$\text{subject to:} \quad \mathbf{x} + \mathbf{r} \in [0, 1]^m, \tag{3.15}$$

where $\text{loss}_f : \mathbb{R}^m \times \{1, \ldots, k\} \to \mathbb{R}^+$ is a continuous loss function associated with the model f. Here, a line search is performed to find a minimum $c > 0$ for which the minimizer $\mathbf{r}$ of the mentioned problem satisfies $f(\mathbf{x}+\mathbf{r}) = l$ [59].

This method for finding adversarial samples creates higher quality samples in a sense that they are closer to the original image, yet misclassified, but is more time-consuming and has much higher computational demands than the previously mentioned FGSM, even though it applies an approximation to the original problem of finding a minimal distortion.

Chapter 4

Experimental Analysis

This chapter introduces an in-depth examination of synthetic image detection. We present a comprehensive exploration of the methodologies, datasets, and experiments conducted to detect synthetic images. The main focus of our investigation was evaluating how well these models can generalize when presented with new previously unseen data, a critical factor in the robustness and reliability of any detection system. In addition, we studied how the degradation of the input images impacts the detection accuracy. Another key area of investigation was the efficiency of adversarial attacks in confusing classification models. Lastly, we dive into the challenging task of localizing fake segments within partially manipulated images.

4.1 Dataset

Our dataset consists of face images produced by five generators; see Fig. 4.1. We use two Stable Diffusion checkpoints: Realistic Vision V5.1 [12] (RV5.1), which is fine-tuned Stable Diffusion V1.5, and official StabilityAI's Stable Diffusion V1.4 [51] (SD1.4). Then three synthetic image sets that are part of the DFFD corpus [15]: FaceApp [19] (FA), which are images produced by a popular commercial mobile phone application with undisclosed technology, and GANs PG-GAN2 [31] (PG), and StyleGAN [32] (SG). Examples produced by the mentioned generators can be seen in Fig. 4.1.

For creating our samples with Stable Diffusion checkpoints, we utilized an open-source project Stable Diffusion Web UI [4], which is a browser interface for Stable Diffusion. There are numerous extensions that can be accessed, including SD Dynamic Prompts extension [18]. This extension implements an expressive template language for the generation of random or combinatorial prompts. In particular, we used the following prompt to increase diversity in our dataset: *"RAW photo, {older | younger} {man | woman | lady | girl | boy} { {smiling | staring} | with glasses | with hat | with {brown | blonde | dark} {straight | curly | short} hair }, high quality portrait taken with Nikon camera, in {nature | a city | a room | an office | a park | a street | a forest }"*. That enabled us to quickly generate diverse images. With different random seeds, we generated almost 1.7k images for both Stable Diffusion checkpoint, RV5.1 and SD1.4. The other generators consist of 2k images for each of FA,

Figure 4.1: Samples of or dataset produced by five generators. Two diffusion models – Realistic Vision 5.1 [12] (RV5.1) and Stable Diffusion 1.4 [51] (SD1.4), one commercial app – FaceApp [19] (FA), two GANs – Progressive GAN [31] (PG) and StyleGAN [32] (SG).

PG, and SG.

For the negative class of real images, we use images from the FFHQ dataset [32]. All synthetic and real images underwent the same preprocessing procedure, cropping with the same margin, aligning using facial landmarks, and resampling to 224×224 px.

4.2 Cross-generator Testing

In this experiment, we trained the ResNET-50 backbone binary classifier [26] to distinguish between synthetic and real samples. The dataset was always split to 80-10-10% for disjoint training-validation-test sets, respectively. The ratio between synthetic and real classes was always 50-50%. We used Adam optimizer with default settings and horizontal flipping as data augmentation. We always selected the model that achieved the best accuracy on the validation set.

We performed the following cross-generator experiment. We first trained on single generator images and tested on all in the set, see Tab. 4.1. Then, the other way around, we trained on all generators with one left out and

		Testing set				
		RV5.1	SD1.4	FA	PG	SG
Training set	RV5.1	**100**	58	49	50	50
	SD1.4	51	**100**	50	54	49
	FA	53	50	**80**	87	60
	PG	49	61	54	**100**	50
	SG	48	48	54	66	**94**

Table 4.1: Cross-generator testing – training on a single generator. Each cell (row, col) shows test accuracy in the percent of the models trained on data from generator *row*, tested on data from generator *col*.

		Testing set				
		RV5.1	SD1.4	FA	PG	SG
Training set	-RV5.1	**58**	92	80	94	91
	-SD1.4	91	**84**	85	91	91
	-FA	95	94	**55**	94	89
	-PG	93	92	77	**79**	85
	-SG	95	95	80	94	**52**

Table 4.2: Cross-generator testing – leave one out training. Each cell (row, col) shows test accuracy in percent of models trained on the whole training set except the data from generator *row*, tested on data from generator *col*.

tested again on all, see Tab. 4.2.

We can observe in Tab. 4.1 that if the detector is trained on the same model as it is tested (diagonal of the table), the accuracy is perfect for RV5.1, SD 1.4, PG, and very high for SG. The accuracy is only 80% for FA. FA, FaceApp [19], a commercial app with unknown technology behind, probably blends the real face with some manipulations, making it harder to identify. However, we can clearly see that accuracy drops close to chance when we test on images produced by generators the detector was not trained on (off the diagonal). Interestingly, this is not the case for FA, which achieves even higher accuracy on PG, which might indicate similar technology, but the converse is not true. The generalization does not occur for even very similar models; the diffusion models RV5.1 and SD 1.4 share the same architecture. The first is a fine-tuned version of the latter.

In Tab. 4.2, when the detector is trained on data multiple generators, a certain level of generalization to unseen generators is achieved for some models, as seen in the diagonal now. SD1.4 seems to generalize well while it was not trained on it. However, RV5.1 is very close to SD1.4, but the generalization is not reciprocal. PG seems to generalize too, and the rest is close to chance.

4.2.1 Adapting to New Generators

We see that detector generalization to an unseen model is a problem. Therefore, in the following experiment, we measure how many samples from the

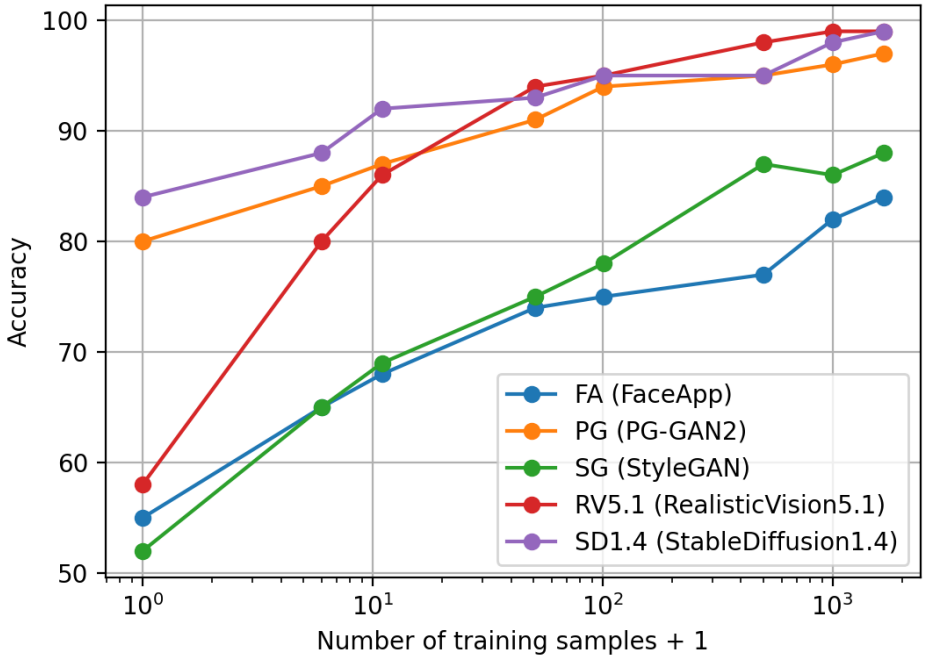


Figure 4.2: Learning curves for training a detector to spot images produced by a new generator. Test accuracy as a function of the number of training samples. The horizontal axis is logarithmic.

new generator are needed for fine-tuning. We always start from the model that is trained on all the generators of our set except one (i.e., the rows of Tab. 4.2), so its initial accuracy is on the diagonal of Tab. 4.2. Then we successively add training samples of the new generator (0, 5, 10, 50, 100, 500, 1000, 1666) samples and measure the accuracy on the test set. The results are shown as learning curves in Fig. 4.2. Note that the plot has a logarithmic horizontal axis. Interestingly, for some generators, only units or small tens of training samples are needed to significantly improve detection accuracy.

4.2.2 Comparison with the State of the Art

We evaluate recent fake image detectors on our test set produced by the RV5.1 generator. We tested Wang 2020 [68], HiFi 2023 [23] which both provide pre-trained models, and Durrall 2019 [17] which we trained on the independent training split of the RV5.1 dataset by using the training script provided by the authors. Our ResNET-50 was trained on the same set.

The results are shown in Tab. 4.3. Although Wang 2020 [68] claimed to generalize to unseen generators, it no longer holds with novel generators. The model achieved accuracy close to chance. It was trained on GAN-based generators which does not provide any generalization to the recent diffusion RV5.1 generator. Moreover, the recent HiFi 2023 [23] also failed, despite being trained on diffusion models. It achieves low accuracy of 44.2%, as it

Model	Accuracy
Wang 2020 [68]	48.3
HiFi 2023 [23]	44.2
Durall 2019 [17]	87.7
ResNET-50 (ours)	99.5

Table 4.3: Comparison with the state of the art. Accuracy on test set produced by RV5.1 generator. The last two models were trained on an independent split of the RV5.1 dataset.

confused almost all RV5.1 images with real ones while classifying some of the real images as synthetic. This generalization failure is particularly striking since this method presented in the last CVPR conference is supposed to be the state of the art. This finding reveals the difficulty of the problem of detecting unseen generators.

Durral 2019 [17] resulted in an inferior accuracy compared to our model. The reason is probably that it uses very simple features, magnitude spectrum radius, and logistic regression.

4.3 Robustness to Image Degradations

We have seen that the discriminator can be trained well to distinguish between real and fake images, given that the generator (or at least its class) is known. In this series of experiments, we studied how various degradations of the input images affect the classification accuracy. Namely, we tested Gaussian blur, JPEG compression, patch size, and downsizing.

We conducted two distinct evaluations. Initially, we assessed the model that had been trained exclusively on undistorted images. Subsequently, to address image degradation, we implemented a second scenario where the detector was trained anew, this time incorporating image degradation into the training process as a form of data augmentation. To ensure significance and avoid results based on a single, potentially anomalous model, we repeated this training process ten times, each time creating a separate model. In the following figures, this is denoted by the error bars that represent the standard deviation.

4.3.1 Gaussian Blur

In this experiment, our aim was to investigate how image blur affects the performance of our classification model. We began with a pre-trained ResNET-50 model, initially trained for clear, unaltered images. To assess the model's resilience to image quality degradation, we subjected our test image set to blurring at ten different intensity levels. Examples can be seen in Fig. 4.3.

Each level of blurring represented a distinct degree of image distortion, ranging from minimal to severe. This gradation allowed us to systematically explore the model's response to progressively degraded images. After applying

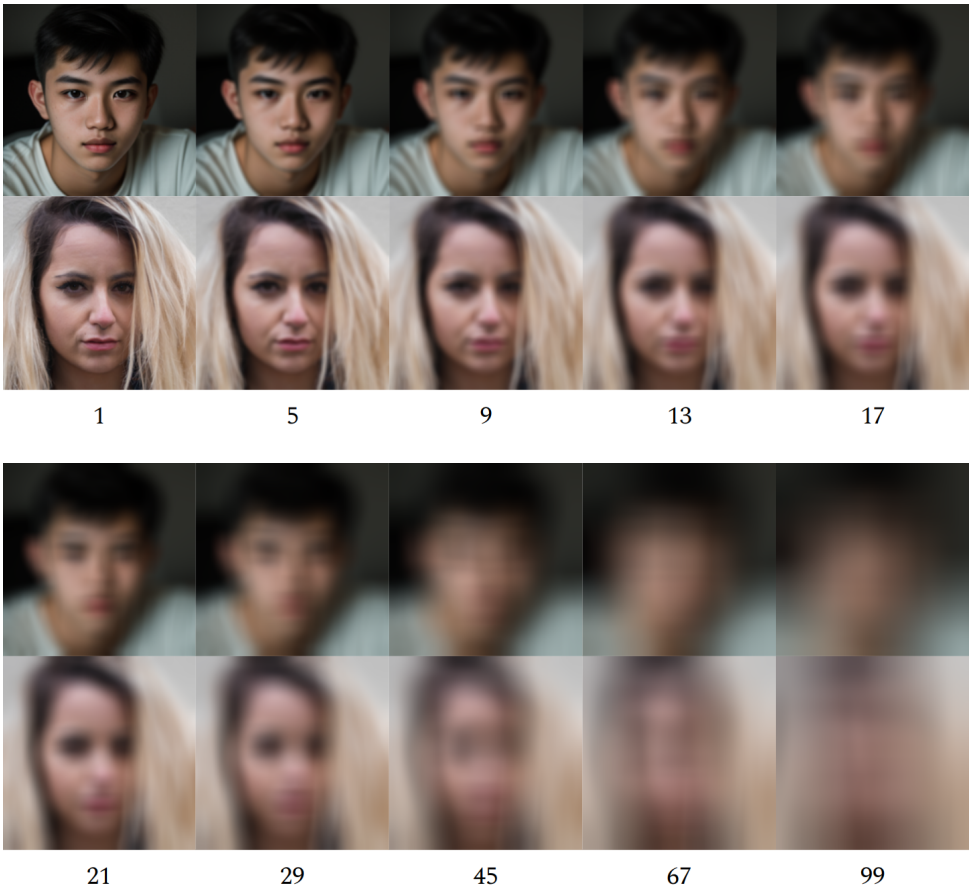


Figure 4.3: Examples showing the Gaussian blur strength. The numbers indicate kernel size σ.

each level of blurring to the test images, we then evaluated the model's classification accuracy.

In the second scenario, instead of applying blur solely to the test images, we introduced blurring as a part of the data augmentation strategy during the model's training. We used the same range of ten blur intensity levels, applying these transformations to the training images. This approach was designed to expose the model to a variety of blurred images during its learning phase, with the intention of enhancing its ability to generalize and maintain accuracy when encountering blurred images during testing.

The results of all the values tested are visible in Fig. 4.4. Interestingly, the model that was pre-trained with blurred images was able to achieve 70% accuracy even if the Gaussian blur was as strong as $\sigma = 21$px. Without pretraining, a similar result was achieved with $\sigma = 9$px.

4.3.2 JPEG Compression

In the following experiment, we investigate the effect of JPEG compression on the accuracy of our real/fake image classification model. JPEG compression

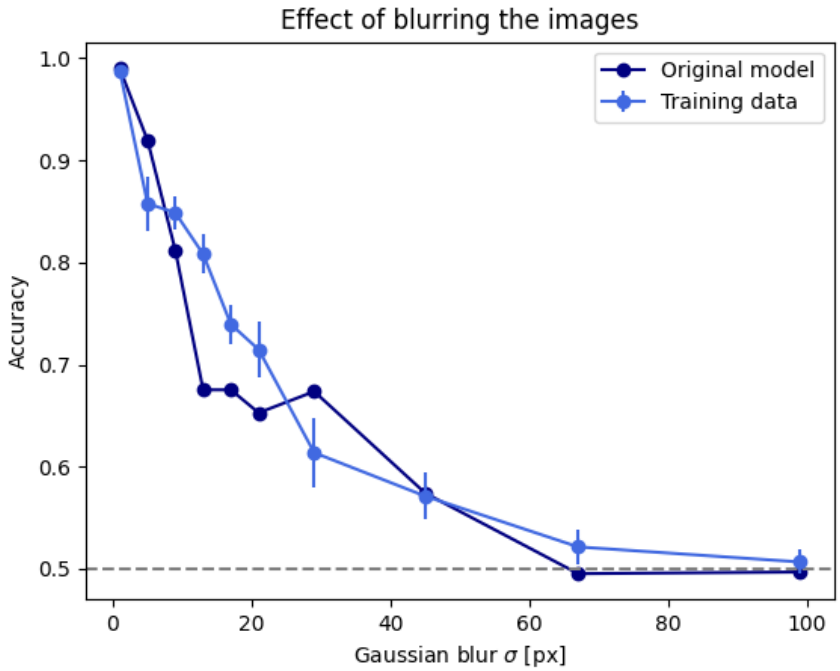


Figure 4.4: Effect of increasing the intensity of the Gaussian blur on the classification accuracy.

introduces artifacts and reduces image quality, a factor that could obscure the subtle distinctions between real photographs and synthetic images of people.

We applied different levels of JPEG compression to both real and artificially generated images, altering the compression quality from high (minimal compression, preserving more detail) to low (maximum compression, with significant loss of detail and introduction of artifacts). See Fig. 4.5 for examples. The objective was to assess whether compression could 'equalize' the visual fidelity between the two sets of images, thereby making it more difficult for the model to classify them correctly. The experiment was designed to simulate conditions where images might be compressed to meet storage or bandwidth constraints, a common occurrence in digital media. The results would indicate how such real-world conditions might influence the performance of models designed to distinguish real images from synthetic ones.

In Fig. 4.6 we can observe that the model is robust against JPEG compression. With JPEG-compressed samples in the training set, it achieves an accuracy about 88% even if the JPEG compression quality is as low as 10. The model's ability to maintain high accuracy under these conditions suggests that it has successfully learned to identify deep, discriminative features that are less sensitive to the kinds of distortions introduced by JPEG compression.

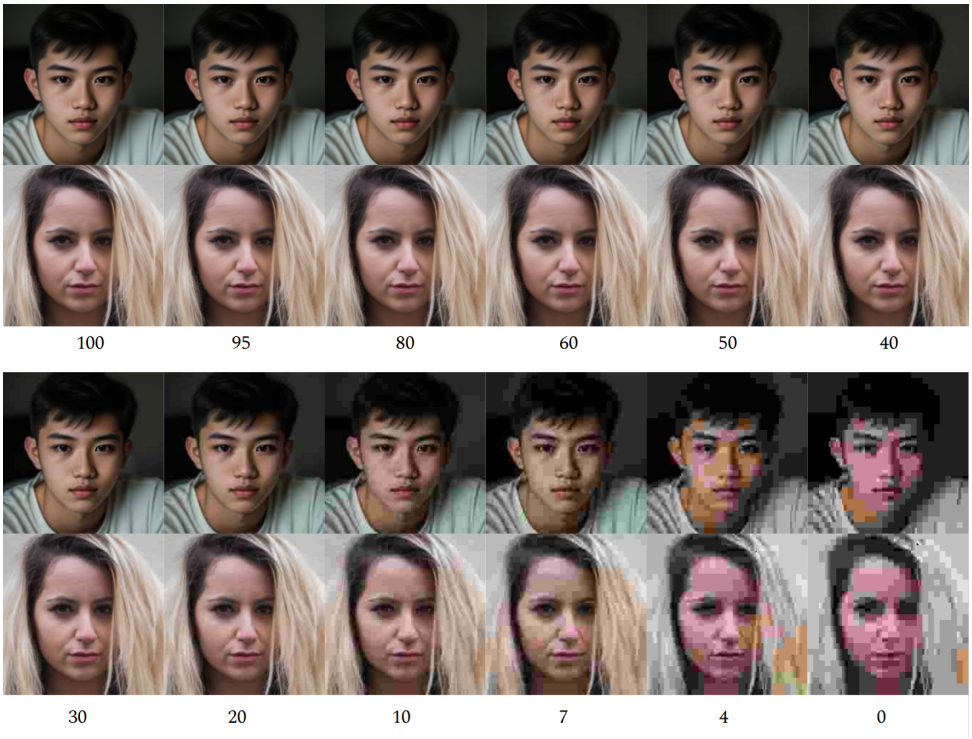


Figure 4.5: Examples showing various levels of JPEG compression quality. Numbers indicate the JPEG quality setting.

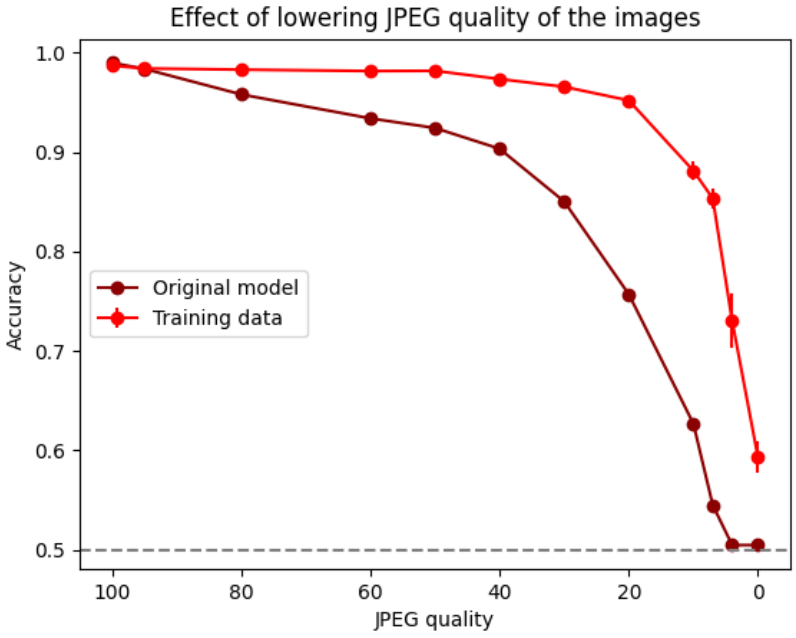


Figure 4.6: Effect of increasing the intensity of the JPEG compression on the classification accuracy.

4.3.3 Patch Size

In this experiment, our primary objective was to determine the minimum portion of an image necessary for an accurate classification by our model. We employed a method in which parts of each image were obscured with black pixels, leaving only a randomly placed square patch visible. This approach was aimed at simulating scenarios where image visibility is inherently limited.

Our images were initially 224×224 pixels in size. We started with the entire image visible and progressively decreased the size of the unmasked, visible patch. See Fig. 4.7 for examples. This gradual reduction allowed us to systematically study how the diminishing field of view affects the model's image classification capabilities.

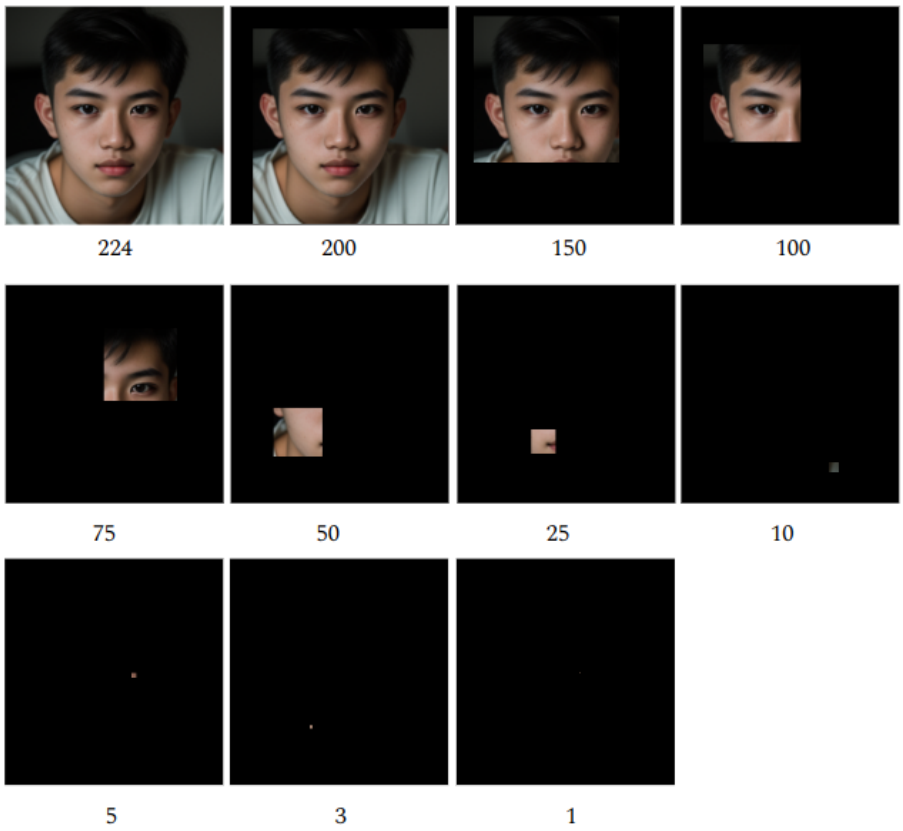


Figure 4.7: Examples showing visible patch sizes. The visible patches are placed randomly. Numbers indicate patch size in pixels.

Each step involved a specific patch size, decreasing incrementally to represent various levels of visibility. By doing so, we could accurately measure the impact of decreasing visibility on the accuracy of the model. This approach was critical in understanding at what point the reduction in the visible area began to significantly hinder the model's ability to classify the images correctly. The results for all the values tested are visible in Fig. 4.8. With the re-trained model, the 70% classification accuracy was achieved with visible patches as small as 25×25 px.

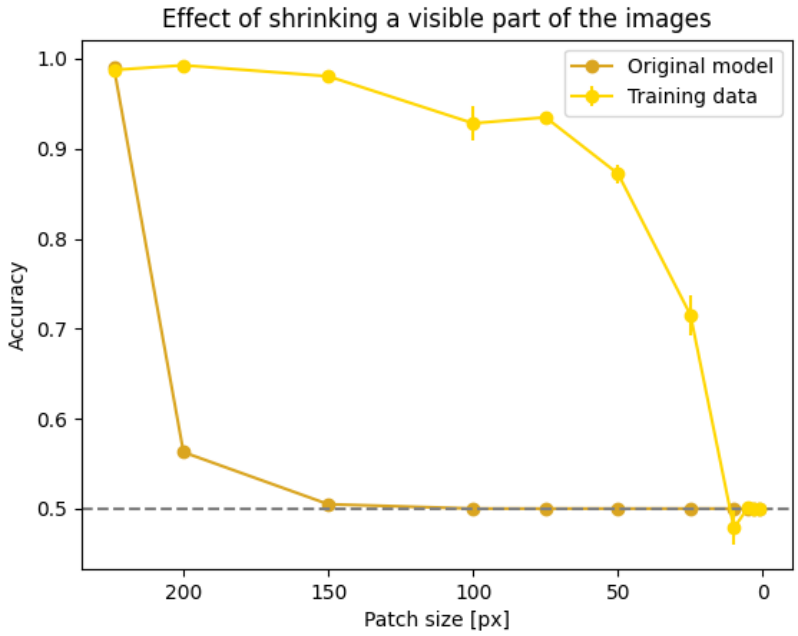


Figure 4.8: Effect of increasing the size of a visible patch on the classification accuracy.

4.3.4 Downsizing

In a series of experiments aimed at testing the resilience of our image classification model, we introduced downsizing as a means to challenge the model's ability to discern real from artificially generated images. The motivation for this type of experiment was that oftentimes the images have the resizing step as one of the input transformations. Therefore, we wanted to simulate a scenario where an image with a smaller resolution than what the model was trained for is passed into the model.

The original size of the images is 224×224 pixels, which is an input receptive field of our model. We resize the image down to a smaller resolution and then resize it back to 224×224 pixels again. The default anti-aliasing filter was used before subsampling and the default bilinear interpolation was used for both downsampling and upsampling. Examples are shown in Fig. 4.9. The results of the experiment can be seen in Fig. 4.10. More than 80% of the accuracy is preserved when the face image is of 50×50 px resolution.

4.3.5 Conclusion

The experiments conducted within this chapter have yielded insightful revelations about the robustness of our image classification model in the context of distinguishing real from synthetic images. Across a series of systematic tests, applying Gaussian blur, adjusting JPEG compression, varying patch visibility and input resolution, the classification model has demonstrated a

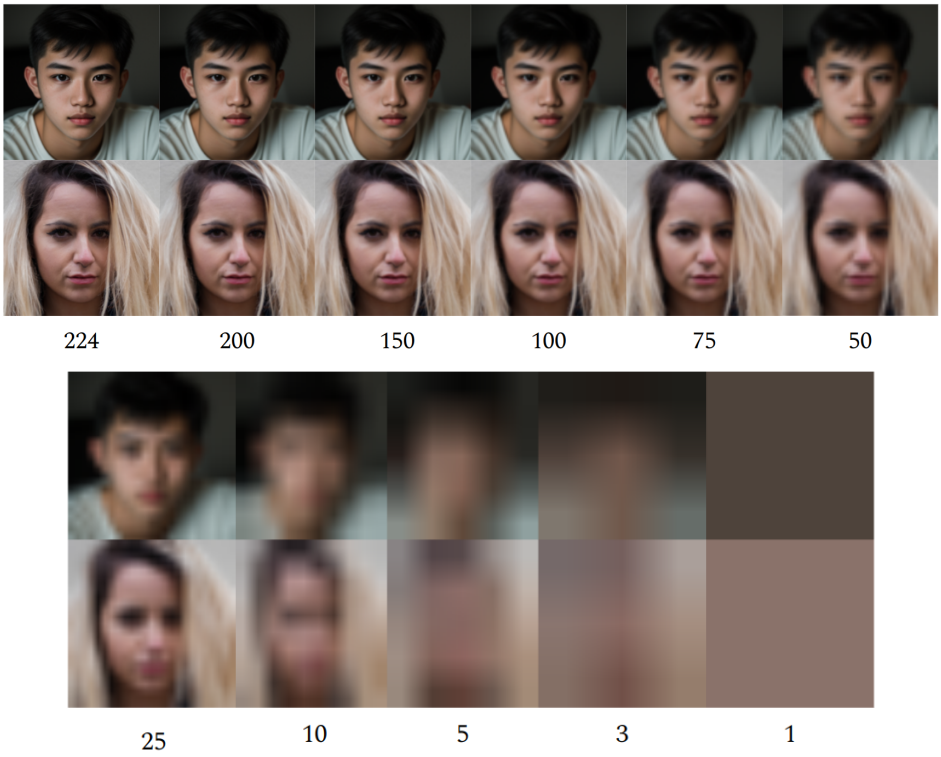


Figure 4.9: Examples showing how downsizing affects image appearance.

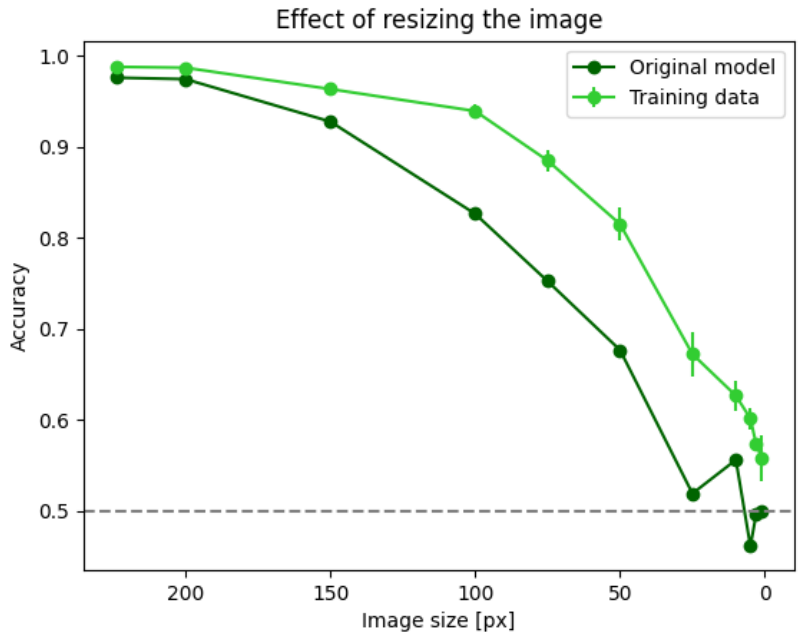


Figure 4.10: Effect of downsizing the image on the classification accuracy.

remarkable degree of resilience to various types of image degradations.

In scenarios where the model was re-trained with degradation as part of the data augmentation, its robustness was enhanced. For example, it maintained an accuracy rate of 80% even with a substantial Gaussian blur ($\sigma = 17$ px), a compelling 90% accuracy with JPEG images of quality as low as 10, and a commendable 70% accuracy when only a small 25×25 pixel patch of the image was visible. These results were not just affirmations of the model's initial robustness, but also that incorporating transformed images into the training data improves the robustness against the degradations.

4.4 Adversarial Attacks

In this section, we study the vulnerability of our fake image detector to adversarial attacks. As described in Sec 3.2, an adversarial attack means performing a hardly perceptible modification of an image (residuum) that causes a change in the classification of the detector, namely, a synthetic image were classified as real. The two methods that we used to obtain the residua, FGSM and L-BFGS, were described in Section 3.2.1 and Section 3.2.2, respectively. We study how adversarial attacks can generalize among training data and model architectures and whether there is an effective defense against it. For this series of experiments, we trained ResNET-50, Xception [11] (another convolutional neural network), and ViT-tiny [63] (visual transformer) on a dataset containing real images and generated images from Realistic Vision 5.1. We then tested whether the residua found for a specific image and a specific detector act adversarially on another image for the same or a different detector model.

We measured the success of attacks by the *confusion rate*, which depicts a percentage of test cases when the model switched the classification due to the attack from "fake" to "real" over the number of "fake" decisions prior to the attack.

In the end, we used only FGSM adversarial attacks, because the L-BFGS method was too time-consuming and with FGSM we could carry out large-scale tests and compare various strengths of attacks by modifying the parameter ϵ. Visual examples of these two methods can be seen in Fig. 4.11 and Fig. 4.12, respectively. The effect of the parameter ϵ on the visual result is shown in Fig. 4.13.

4.4.1 Adversarial Attacks Generalization

Szegedy et al. [59] already study whether adversarial samples are universal – either for different subsets of the training data or among different model architectures. We study similar scenarios with novel model architectures and our data.

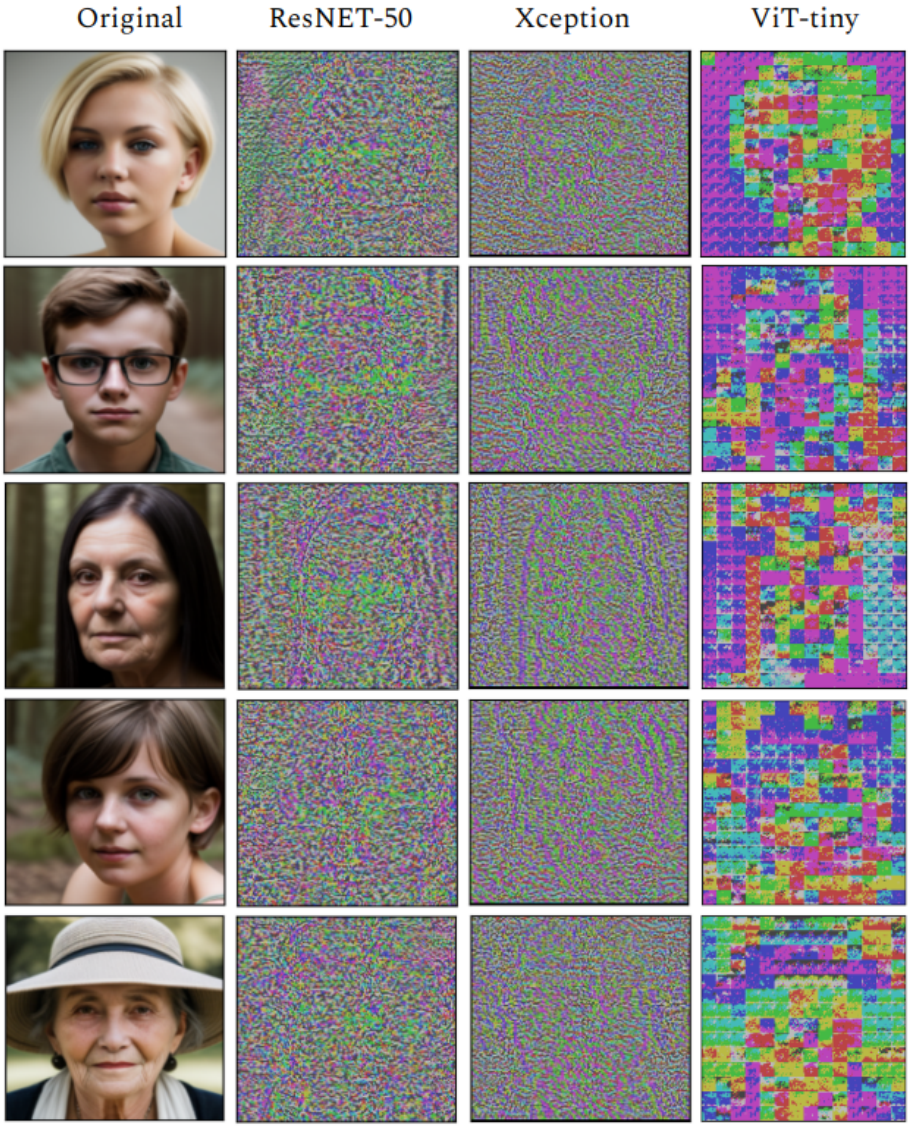


Figure 4.11: Examples of adversarial residua created with FGSM shown for input images and three fake detector models (ResNET-50, Xception, ViT-tiny).

Cross-sample Generalization

Firstly, we wanted to know whether the perturbations (residua) are universal among particular samples. In other words, we found a perturbation for one image that would fool the classifier and apply it to all different images. We used FGSM again due to its speed and performed three experiments with different ϵ settings, as can be seen in Table 4.4. We can observe that the Xception model is the most prone to attacks, as already with $\epsilon = 0.02$ almost all samples are classified incorrectly with the same perturbation. On the other hand, ViT-tiny is more robust against that type of attack if ϵ is low

enough.

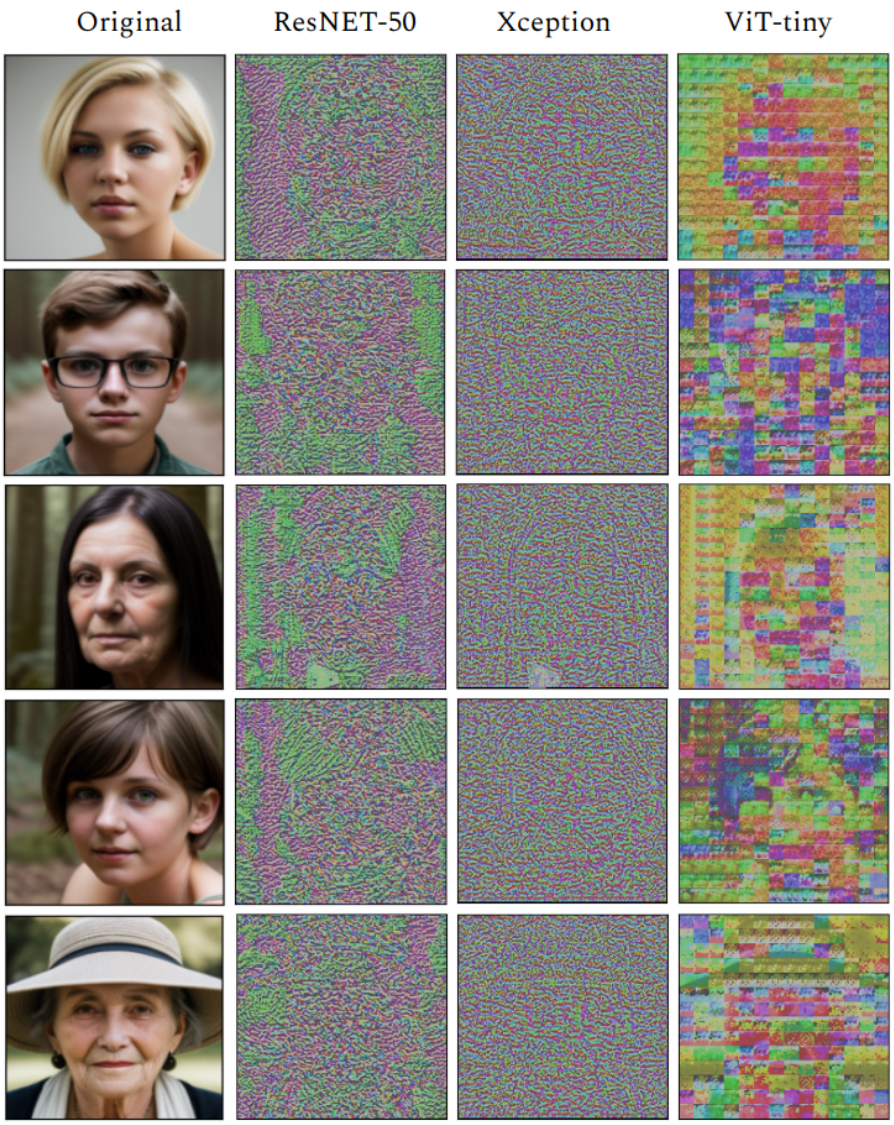


Figure 4.12: Examples of adversarial residua created with L-BFGS shown for input images and three fake detector models (ResNET-50, Xception, ViT-tiny).

We show this experiment as an interesting insight, although this scenario is not so relevant practically, since an attacker would prepare an optimal residuum right for his fake image.

Cross-training-set Generalization

In this scenario, we trained two neural networks on disjoint training sets from the same domain of real and fake images. The fake images were created with the Realistic Vision 5.1 checkpoint. We used classical ResNET-50 and split

	$\epsilon = 0.01$	$\epsilon = 0.02$	$\epsilon = 0.05$
ResNET-50	2.20	67.40	100.00
Xception	96.34	100.0	100.00
ViT-tiny	8.63	25.88	97.25

Table 4.4: Cross-sample adversarial attacks. Adversarial residua found for a given image were tested on different images. The table shows confusion rate for increasing strength of residua ϵ. The experiment was done separately for three model architectures.

	ResNET-50+	ResNET-50-
ResNET-50+	100%	80%
ResNET-50-	84%	100%

Table 4.5: Cross training-set adversarial attacks. The attacks were tailored for a model (column) that was trained on a different dataset subset than the predicting model (row). The table shows confusion rates.

the training data into two disjoint, equally sized subsets. The same testing data are used for both models and the adversarial examples were created with FGSM ($\epsilon = 0.025$). So, referring to Table 4.5, we created the adversarial sample with a model in the column and classified the created sample with a model in a row. We can observe, that a large fraction of examples will be indeed misclassified by the other network, that was trained on a different data. This suggests, that even in a black-box scenario when we do not have access to the weights of the model, we can train a "mirror" model and still have a high confusion rate of adversarial attacks. By the "mirror" model, we mean a model of the same architecture but trained on different data samples.

Cross-architecture Generalization

The most important question that we posed was the following. If we create adversarial samples with one model, are they effective for an adversarial attack on a model with a different architecture? For this experiment, we used two convolutional neural networks – ResNET-50 [26] and Xception [11], and one visual transformer – ViT-tiny [63]. Again, we used FGSM, with three different ϵ values (0.01, 0.02, 0.05). For these three values, we trained each model on the whole Realistic Vision 5.1 dataset and created adversarial samples for these models. Then we run the inference on these perturbed data samples with the other models. The results can be seen in Table 4.6. On the diagonals, there are results of the vanilla adversarial attack, where the adversarial samples were tailored for the model with which the inference was performed. Interestingly, the strength of the FGSM attack quite noticeably altered the results for different architectures. For example, we can see that the FGSM adversarial attack with $\epsilon = 0.01$ is enough for ResNET-50 and Xception, but for ViT-tiny, the attack must be stronger in order to fool the model in all cases. Off the diagonals, we can observe how we can fool the

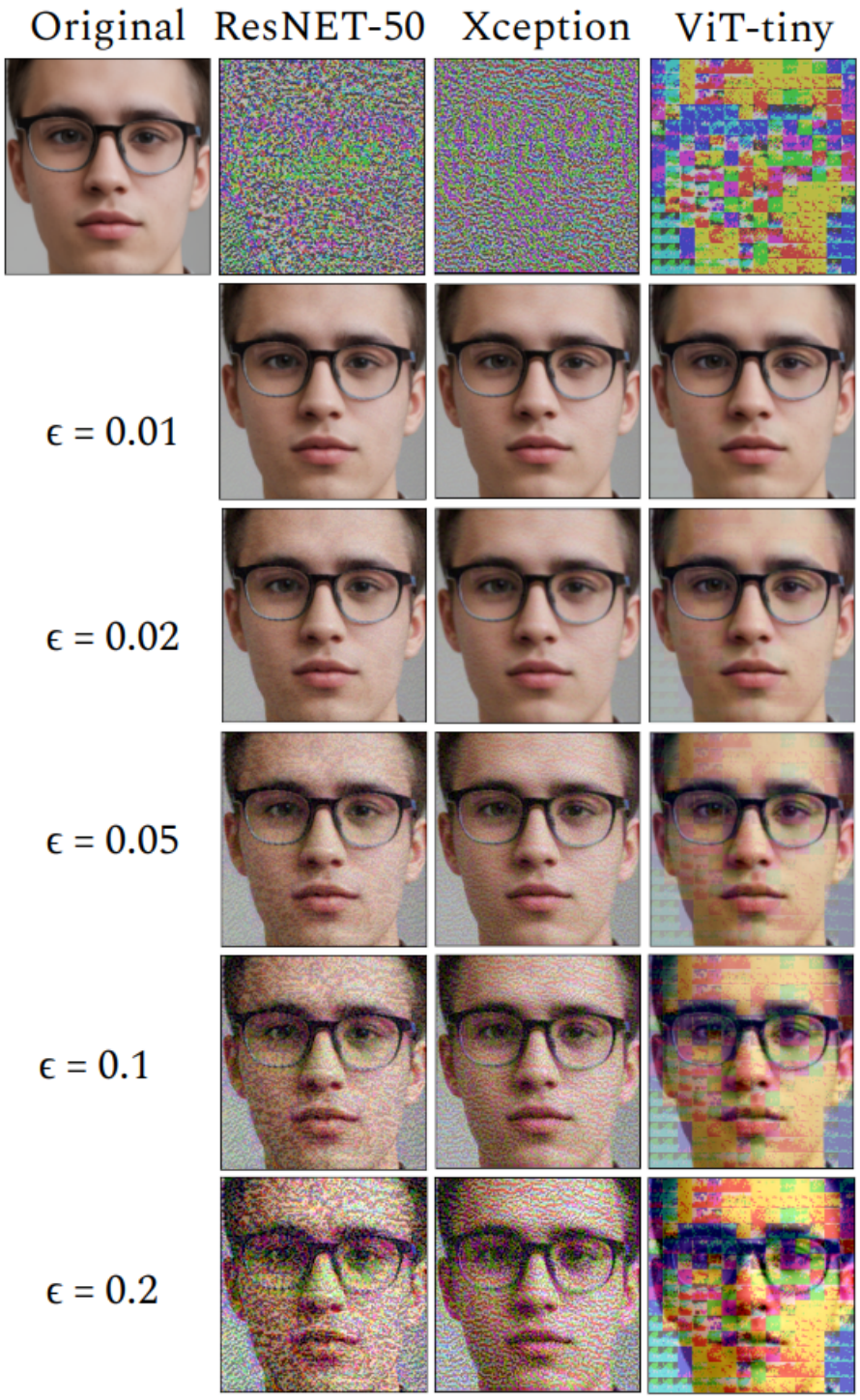


Figure 4.13: Adversarial images crafted to confuse the classification model. They are created by summing the original (top left) image with an adversarial residuum (top row) scaled by ϵ. Results are shown for three models (ResNET-50, Xception, ViT-tiny) with increasing strength ϵ.

model, even if we trained the "mirror" model with a different architecture. As we can see, even these attacks were usually successful.

FGSM $\epsilon = 0.01$	ResNET-50	Xception	ViT-tiny
ResNET-50	**100.00**	67.15	7.30
Xception	2.55	**100.00**	6.93
ViT-tiny	0.39	6.64	**59.77**

FGSM $\epsilon = 0.02$	ResNET-50	Xception	ViT-tiny
ResNET-50	**100.00**	100.00	8.76
Xception	48.54	**100.00**	8.39
ViT-tiny	16.80	89.06	**92.97**

FGSM $\epsilon = 0.05$	ResNET-50	Xception	ViT-tiny
ResNET-50	**100.00**	100.0	15.69
Xception	100.00	**100.0**	10.22
ViT-tiny	100.00	100.0	**100.0**

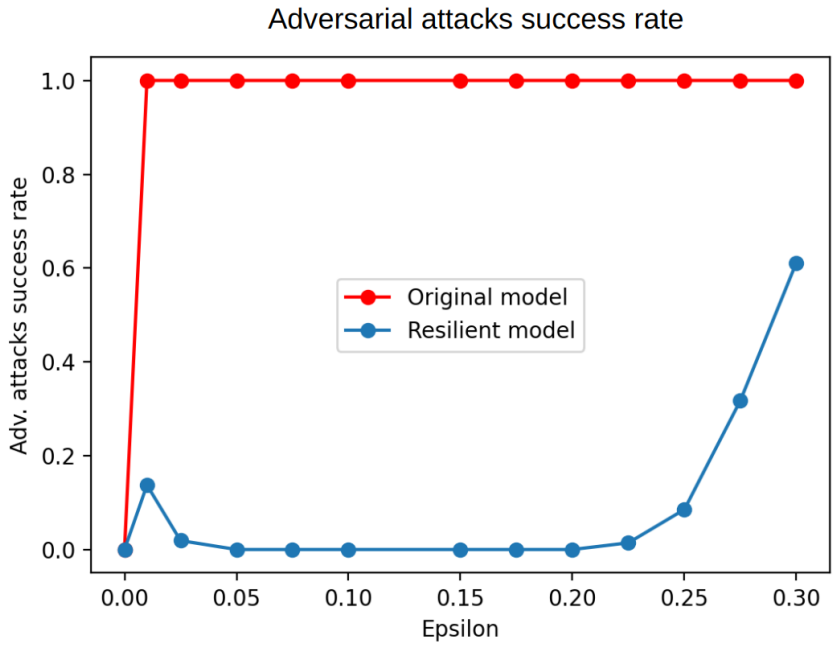
Table 4.6: Cross-architecture adversarial attacks for increasing strength of the residua ϵ. Each cell (row, col) corresponds to confusion rate in percent. The attack was targeted against a model of architecture in *row* and tested against the model of architecture in *col*.

4.4.2 Adversarial Training

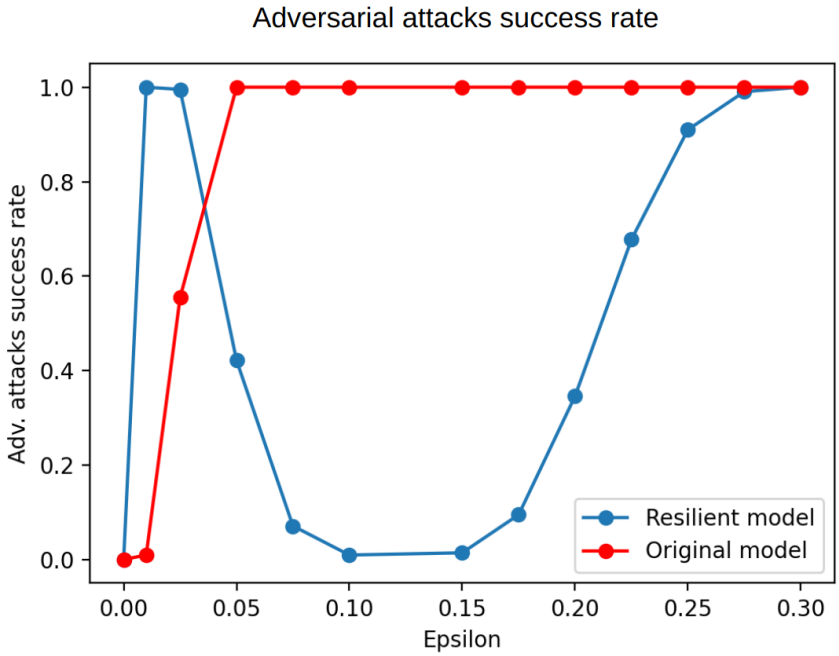
Adversarial training is a defense technique used in machine learning, particularly in the context of deep learning, to improve the robustness of models against adversarial attacks. The core idea behind adversarial training is to expose the model to adversarial examples during the training process so that it learns to correctly classify not only clean, unaltered examples but also those that have been intentionally perturbed by adversarial attacks.

For each image, we created an adversarial sample with FGSM with 6 different strength values ϵ (0.05, 0.1, 0.15, 0.2, 0.25, 0.3) and trained a new ResNET-50 model on this dataset. In the first scenario, we created adversarial attacks for the original model and observed, whether the model trained with the adversarial samples in the training set is more resilient to the attacks. In the second scenario, we created adversarial attacks for the resilient model and observed, whether the adversarial training prevents the adversarial attack. The result of adversarial attacks is depicted in Fig. 4.14.

We tried several strengths of FGSM attacks on the resilient model, namely $\epsilon \in (0.01, 0.25, 0.05, 0.075, 0.1, 0.15, 0.175, 0.2, 0.225, 0.25, 0.275, 0.3)$, so also strengths of adversarial attacks that were not present in the training data



(a) Adversarial samples were created for the original model



(b) Adversarial samples were created for the resilient model

Figure 4.14: Adversarial training. Plots showing how the adversarial attacks were successful. For ϵ value 0, we get a confusion rate of 0, because those are the original images. In scenario (a) the adversarial samples were created for the original model and in scenario (b) for the resilient model. Then we used both the original and the resilient model for predictions.

	Base	Blurred original images	Basic adv. attack	Blurring adv. samples
Accuracy [%]	99	95	3.2	78
Confusion rate [%]	-	-	96	18

Table 4.7: Results of using Gaussian blur ($\sigma = 5$) to mitigate effects of an adversarial attacks. We had a pre-trained ResNET-50 classification and measured accuracy on original unaltered images, blurred images, adversarial samples, and blurred adversarial samples, respectively.

are included. We can observe in Fig. 4.14a that the resilient model that was trained on the adversarial samples is not much vulnerable to attacks anymore. In image Fig. 4.14b, we can see that adversarial training considerably decreases the confusion rate of adversarial attacks, because adversarial samples that were created for the resilient model are only partially successful.

4.4.3 Defense via Input Transformation

Another means of defense against adversarial attacks is somehow modifying the input data, for example by blurring. Many adversarial attacks rely on precise, small changes, usually of a high-frequency nature in the input image, to fool the classifier. By applying low-pass filtration like Gaussian blurring, these crafted perturbations can be disrupted. This can reduce or even nullify the effectiveness of the adversarial attack.

In this experiment, we performed adversarial attacks with FGSM with $\epsilon = 0.05$ on 274 fake images created with Realistic Vision 5.1. As a means of defense, we tested simple Gaussian blur with $\sigma = 5$. In this scenario, we suggest that the attacker does not know about the defense mechanism used, therefore they cannot take that into account. The result can be seen in Table 4.7. The model reached 99% accuracy on the original, unchanged images. When we used a Gaussian blur on the images and then we ran inference on them, the accuracy dropped slightly, to 95%. Adversarial attacks on the original images were successful most of the time, lowering the accuracy of the classification model to 3.2%. Blurring these adversarial samples results in a classification accuracy of 78%, dropping the confusion rate of the adversarial attacks from 96% to 18%. These results indicate that carefully transforming the predicted images can help to protect the model from adversarial attacks without considerably hurting the model performance on the unaltered images.

4.4.4 Conclusion

In summary, the explorations into the sensitivity of fake image detectors to adversarial attacks, as presented in this section, provide interesting insights. Our experimentation with adversarial attacks underscores a critical vulnerability in machine learning models. The fragility of these systems is evident in the high confusion rates of transfer attacks, which demonstrate the ability of adversarial attacks to generalize across different training data and model

architectures. This highlights the inherent vulnerability of these systems.

The experiments demonstrate that adversarial samples crafted with one architecture can compromise the integrity of others, even those with different structural designs. This phenomenon is further complicated by the fact that adversarial perturbations tailored to one sample can affect others, revealing a startling degree of universality among these deceptive inputs.

Adversarial training, while an effective countermeasure in certain contexts, does not offer complete immunity to these attacks. The finding that adversarial examples with perturbation strengths (both below and above the range used in training) can successfully deceive the model suggests that this defense mechanism has limitations. Particularly concerning is the observation that more subtle, and thus less detectable, attacks remain effective, while more pronounced attacks are perceptible but still potentially misleading.

Based on these discoveries, defensive techniques like Gaussian blurring hold potential as input transformation strategies, effectively hindering the effectiveness of adversarial manipulations to a notable extent. Nonetheless, the enduring vulnerability that remains despite these changes suggests that a comprehensive strategy for protection may be required to enhance the robustness of classification models against the ever-changing menace of adversarial attacks.

Moving forward, it is important to continue the development of robust defense mechanisms. This includes not only refining existing strategies but also innovating new methodologies that can adapt to the sophistication of adversarial tactics. Ensuring the reliability and security of image classification models in the context of adversarial challenges remains an important objective for the field of machine learning.

4.5 Localizing Partial Manipulations

In this section, we will look at partial image manipulation, a sophisticated and increasingly prevalent form of image tampering where only a specific region of an image is altered or fabricated. In the following experiment, we will show that these partial manipulations are easy to identify together with localizing the area of the manipulations.

We first prepared a dataset of partially manipulated face images. We randomly sampled real faces from the FFHQ dataset [32] and, for each image, uniformly changed either the eyes, eyebrows, nose, mouth, or whole face. These regions were detected using facial landmarks [35], and the modifications were implemented using state-of-the-art inpainting techniques provided by Stable Diffusion inpainting [51]. This way we produced a dataset of 3.2k partially manipulated images that were mixed with 540 real images.

The data set was divided into training, validation, and test subsets with proportions of 80%, 16%, and 4%, respectively.

Firstly, we tried to localize the inpainted areas with a simple U-Net [52] model. Quantitatively, it has achieved a test dice score of 58%. Visual results can be seen in Fig. 4.15. The results we obtained with this model were not

Figure 4.15: Results of localization of fake areas in the images with U-Net model. The red areas are incorrect predictions of the fake areas (false positives), green areas are ground truth fake areas undetected by our model (false negatives), and yellow areas are where our model correctly localized the fake areas (true positives).

sufficient. One idea, of why that happened might be that we trained the model on binary masks where the foreground corresponds to the manipulated area, so it was a semantic segmentation task. This labeling does not force the area to be compact. Then, we trained YOLOv8s-seg [64] on the same dataset, which is a YOLO-based architecture [48] with a segmentation head. This was an instance segmentation task, where we converted the binary masks into a list of polygons in YOLO annotations format by finding the contours of the foreground areas in the binary masks.

Qualitative results on the test set are shown in Fig. 4.16. It can be seen that detected regions are found precisely, despite the fact that the manipulated (synthetic region) is sometimes fairly small with respect to the entire (real) image and no obvious artifacts are visible in the images.

Quantitatively, the detector achieved mAP50 98% (mean average precision for 50% prediction/ground-truth detection overlap by intersection over the union). Pixelwise recall and precision were 95% and 91%, respectively.

We compared the detector with HiFi [23] which is supposed to provide localization of the manipulation. However, this detector failed completely and always recognized all our partially manipulated images as real. This again confirms, similarly to our findings in Sec. 4.2, that generalization to localize partial manipulations when using unseen generator models is very challenging.

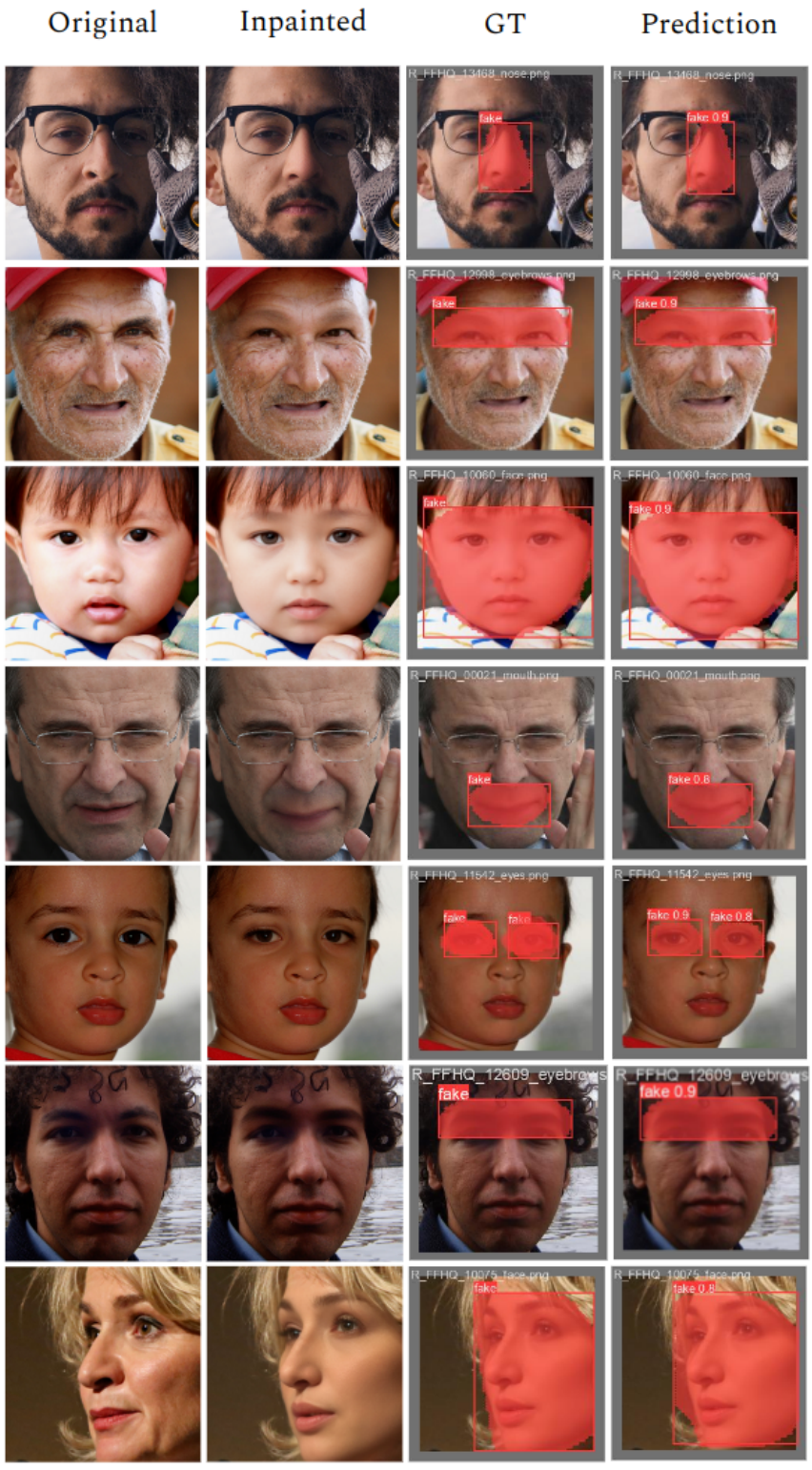


Figure 4.16: Localizing partial image manipulations perpetrated by inpainting of the ground-truth (GT) regions for examples of the test set. Localization predictions were found by our YOLOv8-based model.

All	Q1	Q2	Q3	Q4
0.72	0.24	0.74	0.90	0.97

Table 4.8: Localization accuracy as a function of manipulated area size – mAP50 on the test images. Overall mAP50 is 72% on this dataset, but we can observe, that the model has more trouble localizing the smaller areas (Q1) compared to the larger ones (Q4).

4.5.1 Effects of Manipulated Area Size on Classification Accuracy

Secondly, we quantify how the size of the manipulated area affects the performance of our YOLOv8-seg segmentation model. Smaller manipulations might be more challenging to detect, posing a greater threat to the integrity of visual information.

We created a dataset consisting of 4.5k images with partial manipulations. The manipulated areas were generated by randomly placed, rotated, and cropped ellipse in each image and then we used Stable Diffusion's inpainting to modify the images in these areas. Several examples can be seen in Fig. 4.18. The model was trained on 2.7k of the 4.5k images, 1.8k were used for validation and testing. Firstly, we converted the binary masks to YOLO format annotations, and then we trained YOLOv8 to localize the modified areas. We wanted to demonstrate, that it is more difficult to localize smaller areas. We divided the test data into four equally sized sets based on the areas of the manipulated images: 'Q1': (0, 0.08), 'Q2': (0.08, 0.16), 'Q3': (0.16, 0.25), 'Q4': (0.25, 0.36), where the numbers denote a ratio of the mask areas of the generated parts with respect to the total area. The distribution is denoted in Fig. 4.17. The results are shown in Table 4.8. We can see that it is easier to localize larger areas for the model. Unlike in previous experiments, where we modified facial features (face, eyes, eyebrows, nose, mouth), here we chose the unpainted regions completely randomly. It happens that especially small regions are located in flat areas without texture. These regions do not manifest much of a usable signal for identification, whereas larger inpainted areas are more likely to exploit the natural face symmetry. This can be one of the reasons why the smaller modified areas are harder to localize by the model.

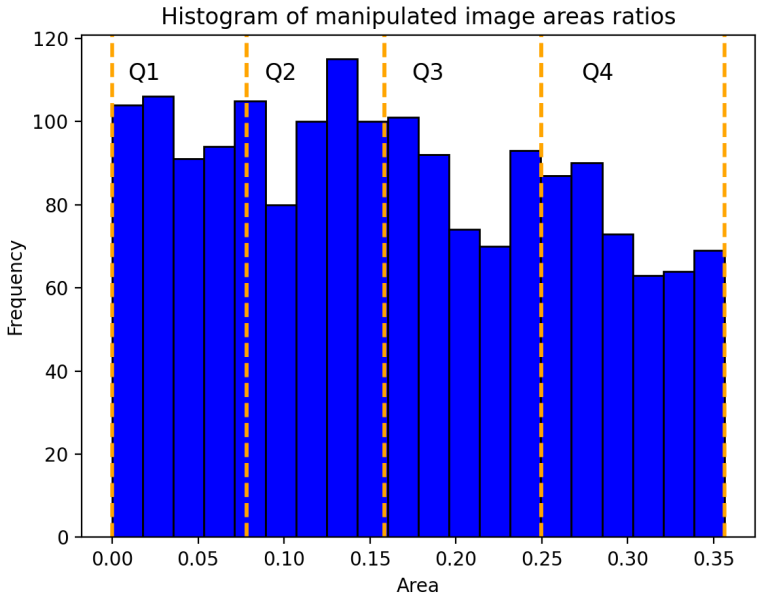


Figure 4.17: Distribution of the test images based on the size of the manipulated area.

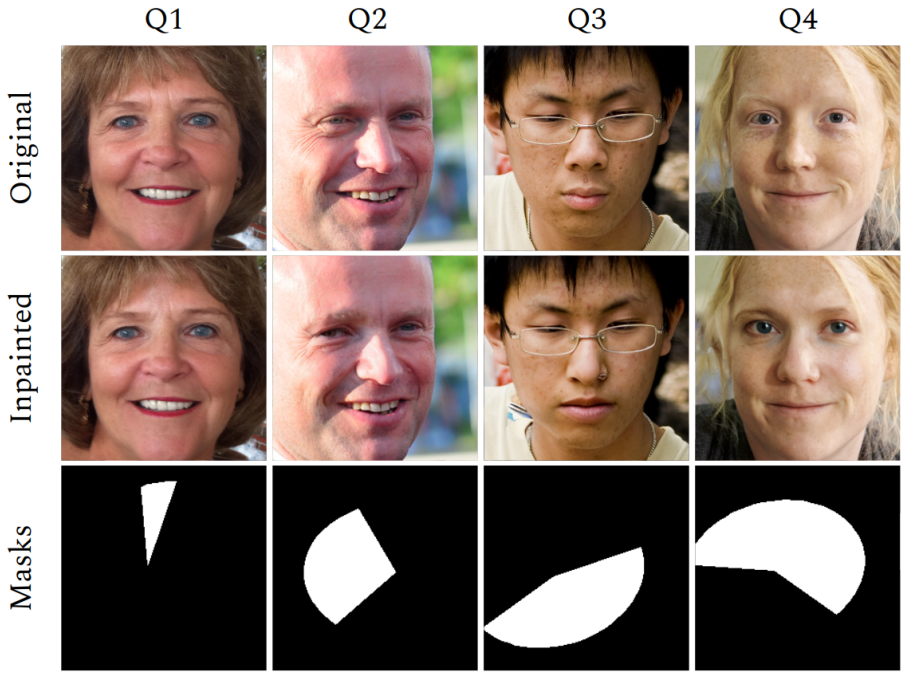


Figure 4.18: Examples of partial image manipulations with random masks. The regions of the ground-truth masks were modified by inpainting. One image from each size quartile is shown. The left-most column has the smallest inpainted region, while the right-most one has the largest one.

Chapter 5

Conclusion

In this work, we studied the detection of synthetic face images and performed several experiments with the following results and conclusions.

The *good news* is that it is possible to train a simple model with an off-the-shelf architecture, which has almost perfect accuracy in distinguishing between synthetic and real images in case the generator of synthetic images is known and available. This holds also for high-quality generators such as StyleGAN or Stable Diffusion and their improved variations, which are hardly distinguishable for a human observer. The accuracy achieved far outperforms human abilities [57]. Another positive aspect is that the detector can be trained with data augmentation, to make it robust to common image distortions (blur, compression, downsizing), and it can achieve good accuracy with only a small input patch from the face. Moreover, it is easy to detect the case of partial manipulations, where a collage of real and synthetic images is made. The manipulated area is automatically localized by training a standard YOLO [64] model.

However, there are also *bad news*. It is easy to prepare an adversarial attack even with a simple method like FGSM. It turns out that the residua found for a target model act adversarially on a model with the same architecture trained on different datasets and even on other models of different architectures. We showed that adversarial images found for vision transformers often confuse convolutional networks. There are some methods that mitigate adversarial attack effectiveness, but if the attacker knows about them, they can take a step ahead and create new adversarial samples. Another bad news is that the detectors do not generalize well to generators they were not trained on. This is not just the case of our simple detector, but we showed that two tested state-of-the-art detectors failed completely and could not detect synthetic images generated by a newer generator, which they were not trained on. This bad news demonstrates that universal detection of synthetic/fake images is a very tough problem, which is far from being solved despite active research in recent times.

Bibliography

[1] Z. Akhtar. Deepfakes generation and detection: A short survey. *Journal of Imaging*, 9(1):18, 2023.

[2] J. Alammar. The illustrated Stable Diffusion, 2022. `https://jalammar.github.io/illustrated-stable-diffusion/`.

[3] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 214–223. PMLR, 06–11 Aug 2017.

[4] AUTOMATIC1111. Stable Diffusion web UI, 2023. `https://github.com/AUTOMATIC1111/stable-diffusion-webui`.

[5] C. Bartz, H. Raetz, J. Otholt, C. Meinel, and H. Yang. Synthesis in style: Semantic segmentation of historical documents using synthetic data. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 3878–3884. IEEE, 2022.

[6] Bloomberg news. The next wave of scams could be deepfake video calls from your boss, Aug 30, 2023. `https://www.linkedin.com/pulse/next-wave-scams-could-deepfake-video-calls-from-your-boss/`.

[7] L. Chai, D. Bau, S.-N. Lim, and P. Isola. What makes fake images detectable? Understanding properties that generalize. In *Proc. ECCV*, 2020.

[8] M. Chen, J. Fridrich, M. Goljan, and J. Lukás. Determining image origin and integrity using sensor noise. *IEEE Transactions on information forensics and security*, 3(1):74–90, 2008.

[9] R. J. Chen, M. Y. Lu, T. Y. Chen, D. F. Williamson, and F. Mahmood. Synthetic data in machine learning for medicine and healthcare. *Nature Biomedical Engineering*, 5(6):493–497, 2021.

[10] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of*

the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2818–2829, 2023.

[11] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proc. CVPR*, 2017.

[12] CivitAI. Realistic vision, v5.1, 2023. `https://civitai.com/models/4201/realistic-vision`.

[13] R. Corvi, D. Cozzolino, G. Poggi, K. Nagano, and L. Verdoliva. Intriguing properties of synthetic images: from generative adversarial networks to diffusion models. In *Proc. CVPR*, pages 973–982, 2023.

[14] R. Corvi, D. Cozzolino, G. Zingarini, G. Poggi, K. Nagano, and L. Verdoliva. On the detection of synthetic images generated by diffusion models. In *Proc. ICASSP*, 2023.

[15] H. Dang, F. Liu, J. Stehouwer, X. Liu, and A. Jain. On the detection of digital face manipulation. In *Proc. CVPR*, 2020.

[16] Y. Dong, H. Su, B. Wu, Z. Li, W. Liu, T. Zhang, and J. Zhu. Efficient decision-based black-box adversarial attacks on face recognition. In *Proc. CVPR*, 2019.

[17] R. Durall, M. Keuper, F.-J. Pfreundt, and J. Keuper. Unmasking deepfakes with simple features. *arXiv preprint arXiv:1911.00686*, 2019.

[18] A. Eyal. Stable Diffusion Dynamic Prompts extension, 2023. `https://github.com/adieyal/sd-dynamic-prompts`.

[19] FaceApp Technology Limited. Faceapp, 2023. `https://www.faceapp.com/`.

[20] H. Farid. Image forgery detection. *IEEE Signal processing magazine*, 26(2):16–25, 2009.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[22] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *Proc. ICLR*, 2015.

[23] X. Guo, X. Liu, Z. Ren, S. Grosz, I. Masi, and X. Liu. Hierarchical fine-grained image forgery detection and localization. In *Proc. CVPR*, 2023.

[24] J. Hancock and J. Bailenson. The social impact of deepfakes. *Cyberpsychology, Behavior, and Social Networking*, 24:149–152, 03 2021.

[25] J. Hao, Z. Zhang, S. Yang, D. Xie, and S. Pu. Transforensics: image forgery localization with dense self-attention. In *Proc. ICCV*, 2021.

[26] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016.

[27] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[28] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.

[29] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris. GANSpace: Discovering interpretable GAN controls. In *Proc. NeurIPS*, 2020.

[30] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[31] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[32] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proc. CVPR*, 2019.

[33] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020.

[34] J. Kietzmann, L. W. Lee, I. P. McCarthy, and T. C. Kietzmann. Deepfakes: Trick or treat? *Business Horizons*, 63(2):135–146, 2020.

[35] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[36] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

[37] T. Klinghoffer, J. Philion, W. Chen, O. Litany, Z. Gojcic, J. Joo, R. Raskar, S. Fidler, and J. M. Alvarez. Towards viewpoint robustness in bird's eye view segmentation, 2023.

[38] Y. Lai, Z. Luo, and Z. Yu. Detect any deepfakes: Segment anything meets face forgery detection and localization. In *Proc. CCBR*, 2023.

[39] D. Lee. Deepfakes porn has serious consequences. BBC News, Feb 3, 2018. `https://www.bbc.com/news/technology-42912529`.

[40] D. Mahapatra and B. Bozorgtabar. Progressive generative adversarial networks for medical image super resolution. *ArXiv*, abs/1902.02144, 2019.

[41] A. Mary and A. Edison. Deep fake detection using deep learning techniques: A literature review. In *Proc. IEEE International Conference on Control, Communication and Computing (ICCC)*, 2023.

[42] M. Meaker. Slovakia's election deepfakes show AI is a danger to democracy. Wired, Oct 3 2023. `https://www.wired.co.uk/article/slovakia-election-deepfakes`.

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

[44] Mohammed. What are Stable Diffusion models & how to use different models in Automatic1111. Medium, 2023. `https://medium.com/javarevisited/what-are-stable-diffusion-models-how-to-use-different-models-in-automatic1111-13c7874ea284`.

[45] T. T. Nguyen, Q. V. H. Nguyen, D. T. Nguyen, D. T. Nguyen, T. Huynh-The, S. Nahavandi, T. T. Nguyen, Q.-V. Pham, and C. M. Nguyen. Deep learning for deepfakes creation and detection: A survey. *Computer Vision and Image Understanding*, 223:103525, 2022.

[46] N. Petrželková. Face image editing in latent space of generative adversarial networks, June 2021. Bachelor's thesis, CTU in Prague, `https://dspace.cvut.cz/handle/10467/94408`.

[47] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[48] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proc. CVPR*, 2016.

[49] Reuters. Erdogan rival accuses Russia of 'deep fake' campaign ahead of presidential vote, May 2023. `https://www.reuters.com/world/middle-east/erdogan-rival-accuses-russia-deep-fake-campaign-ahead-presidential-vote-2023-05-12/`.

[50] A. Rocha, W. Scheirer, T. Boult, and S. Goldenstein. Vision of the unseen: Current trends and challenges in digital image and video forensics. *ACM Computing Surveys (CSUR)*, 43(4):1–42, 2011.

[51] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. CVPR*, 2022.

[52] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[53] E. Schonfeld and A. Veeravagu. Demonstrating the successful application of synthetic learning in spine surgery for training multi–center models with increased patient privacy. *Scientific Reports*, 13(1):12481, Aug 2023.

[54] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proc. ICCV*, 2017.

[55] T. S. Silva. A short introduction to generative adversarial networks. Github, 2017. `https://sthalles.github.io/intro-to-gans/`.

[56] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[57] K. Somoray and D. J. Miller. Providing detection strategies to improve human detection of deepfakes: An experimental study. *Computers in Human Behavior*, 149:107917, 2023.

[58] Y. Song and S. Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

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

[60] D. Tantaru, E. Oneata, and D. Oneata. Weakly-supervised deepfake localization in diffusion-generated images. In *Proc. WACV*, 2024. To Appear.

[61] The Guardian. European politicians duped into deepfake video calls with mayor of Kyiv, Jun 25, 2022. `https://www.theguardian.com/world/2022/jun/25/european-leaders-deepfake-video-calls-mayor-of-kyiv-vitali-klitschko`.

[62] Y. Tian, L. Fan, P. Isola, H. Chang, and D. Krishnan. StableRep: synthetic images from text-to-image models make strong visual representation learners, 2023.

[63] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[64] Ultralytics. Yolov8. GitHub repository, 2023. https://github.com/ultralytics/ultralytics.

[65] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[66] J. Wakefield. Deepfake presidents used in Russia-Ukraine war. BBC News, March 18, 2022. https://www.bbc.com/news/technology-60780142.

[67] C. Wang. Data standardization for robust lip sync. *arXiv preprint arXiv:2202.06198*, 2023.

[68] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros. CNN-generated images are surprisingly easy to spot...for now. In *Proc. CVPR*, 2020.

[69] Z. Wang, Q. She, and T. E. Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[70] X. Wei, S. Liang, N. Chen, and X. Cao. Transferable adversarial attacks for image and video object detection. In *Proc. International Joint Conference on Artificial Intelligence*. AAAI Press, 2019.

[71] L. Weng. What are diffusion models? Jul 2021. https://lilianweng.github.io/posts/2021-07-11-diffusion-models/.

[72] E. Williamson. Q&A: With Zelenskyy surrender hoax, the feared future of deepfakes is here. UVAToday, Mar 2022. https://news.virginia.edu/content/qa-zelenskyy-surrender-hoax-feared-future-deepfakes-here.

[73] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[74] M. Zanardelli, F. Guerrini, R. Leonardi, and N. Adami. Image forgery detection: a survey of recent deep-learning approaches. *Multimedia Tools and Applications*, 82(12):17521–17566, 2023.

[75] J. Zhao, M. Mathieu, and Y. LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.

[76] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.