POLITECNICO DI MILANO

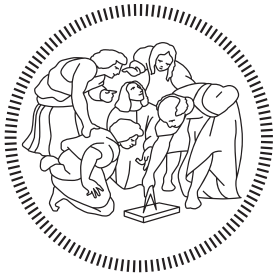
Facoltà di Ingegneria

Scuola di Ingegneria Industriale e dell'Informazione

Dipartimento di Elettronica, Informazione e Bioingegneria

Master of Science in

Computer Science and Engineering



Inpainting based strategies against adversarial attacks

Advisor: PROF. MATTEO MATTEUCCI

Co-advisor: STEFANO SAMELE

Master Graduation Thesis by:

GIORGIO UGHINI

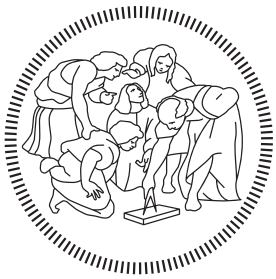
Student Id n. 10535520

Academic Year 2020-2021

POLITECNICO DI MILANO

Facoltà di Ingegneria
Scuola di Ingegneria Industriale e dell'Informazione
Dipartimento di Elettronica, Informazione e Bioingegneria

Corso di Laurea Magistrale in
Computer Science and Engineering



Le strategie di inpainting come difesa contro gli attacchi avversari

Relatore: PROF. MATTEO MATTEUCCI

Correlatore: STEFANO SAMELE

Tesi di Laurea Magistrale di:

GIORGIO UGHINI
Matricola n. 10535520

Anno Accademico 2020-2021

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

http://code.google.com/p/classicthesis/

Happy users of `classicthesis` usually send a real postcard to the author, a collection of postcards received so far is featured here:

http://postcards.miede.de/

This template has been adapted by Emanuele Mason, Andrea Cominola and Daniela Anghileri: *A template for master thesis at DEIB*, June 2015. This version of the template has been later adapted by Marco Cannici, March 2018.

RINGRAZIAMENTI

Vorrei mostrare la mia gratitudine ed il mio affetto verso alcune persone che sono state fondamentali sia nel mio percorso accademico sia nella mia vita, e che hanno contribuito a rendermi la persona che sono oggi.

Come indispensabile premessa, umile nei modi ma di grandi intenzioni, ringrazio il Prof. Matteo Matteucci per avermi scelto per questo lavoro ed avermi permesso di accettare la sfida di studiare un ambito ancora così poco analizzato nella letteratura: le sue ampie vedute e i suoi preziosi consigli sono stati fondamentali per il successo di questo lavoro. Ringrazio Stefano che con passione e sapienza mi ha costantemente seguito nella realizzazione di questa Tesi, fornendomi essenziali consigli scientifici e umani. Egli è stato in grado di indirizzare sapientemente il lavoro nella giusta direzione fornendomi un feedback costante e prezioso.

Vorrei ringraziare Federica per il suo inestimabile supporto nei momenti più difficili, capace di darmi motivazione e forza ma allo stesso tempo insostituibile nello stimolare il mio pensiero critico nei momenti in cui era più necessario. Il suo impegno e aiuto sono tutt'ora senza prezzo ed è anche grazie a lei se oggi mi sento una persona curiosa, attiva ed orientata al capire il perchè delle cose.

Un grazie speciale anche ai miei genitori Tina e Massimo per aver cresciuto la persona che sono oggi. Le basi che mi hanno dato, insieme alla loro morale e al loro supporto economico sono stati un fattore critico nel determinare il successo del mio percorso di studi. Grazie anche alla mia famiglia, in particolare ai miei nonni che seppur di un'altra epoca sono sempre stati in grado di capire l'importanza della formazione scientifica al giorno d'oggi.

Sicuramente affrontare questi cinque anni non sarebbe stato lo stesso senza tutti i miei compagni del Poli, che hanno reso le giornate in aula uniche nel loro genere e senza i quali raggiungere questo obiettivo non avrebbe avuto lo stesso sapore. Grazie ad Andrea, Cosimo e Simeone per essere stati compagni di risate e di studio e per tutte le cose che ho imparato da voi.

Infine, un sincero ringraziamento va anche a tutte quelle persone che in modi diversi hanno condizionato i miei ultimi anni e non solo. Grazie a Elena per essere stata con i suoi traguardi e la sua determinazione fonte per me di enorme motivazione, grazie a Enrico e Riccardo per le nostre serate e i nostri discorsi fondamentali per staccare e ricaricare le batterie, grazie ai miei professori del Politecnico per avermi insegnato così tanti concetti e a quelli del Liceo per avermi trasmesso la passione per le scienze matematiche e informatiche.

Dedico questa Tesi a tutti voi.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The last few years have witnessed dramatic improvements in Machine Learning which led to major breakthroughs in many fields including computer vision, speech recognition, and time series forecasting. These achievements, which often outperform human beings, are based on powerful techniques and algorithms which are all grouped in the field named "Deep Learning".

However, Machine learning systems are not inherently secure. The great results of Deep Learning has led to the investigation of possible vulnerabilities, finding that those models are susceptible to adversarial attacks which dramatically reduce their performance. Adversarial attacks are a method to craft slightly perturbed inputs which exploit the multi-dimensional boundary shape of the model in order to fool it, without compromising the perception that human beings have of it. Intuitively, adversarial inputs look totally different from the real inputs from the eyes of neural networks despite being similar to humans.

Since the discovery of adversarial attacks of neural networks, a plenty of countermeasures and detection techniques have been proposed but reliable defenses to these attacks are an unsolved challenge.

In this Thesis we present Defense by Massive Inpainting, a novel approach to harden neural network image classifier from adversarial attacks based on a zero-trust strategy of the input. More specifically, it performs a total erase of the input image and reconstruct it from scratch thanks to the inpainting technique.

Experiments on this model shown the improved accuracy that it has against the harder type of such attacks, namely the white box adversarial attacks. Moreover, we compare the results of three different attacks under three different networks against a specific baseline chosen because it is the only using inpainting to restore parts of the images to protect from adversarial attacks.

We also propose an evolution of such baseline to allow a faster inference procedure with no loss on accuracy by changing the way in which it selects the areas to inpaint. We called this second framework Saliency Based Localization and Inpainting. Still, this new model is not as successful as Defense by Massive Inpainting in being resilient to adversarial attacks and does not follow a strict zero-trust approach.

This result confirms that the robustness of Defense by Massive Inpainting is due to its zero-trust approach, which does not believe in any information embedded in the adversarial image without firstly purify it.

SOMMARIO

Negli ultimi anni il Machine Learning ha beneficiato di notevoli miglioramenti che hanno portato a importanti scoperte in molti campi, tra cui la visione artificiale, il riconoscimento vocale e la previsione delle serie temporali. Questi risultati, che spesso superano le prestazioni degli esseri umani, si basano su potenti tecniche e algoritmi appartenenti al "Deep Learning".

Tuttavia, i sistemi di Machine Learning non sono intrinsecamente sicuri. Infatti, le ottime prestazioni del Deep Learning hanno portato a indagare su possibili vulnerabilità ed è stato scoperto che questi modelli sono suscettibili agli attacchi avversari, tecniche che permettono di ridurne drasticamente le prestazioni. Gli attacchi avversari sono un insieme di tecniche per creare input leggermente perturbati che ingannano la rete neurale, senza compromettere la percezione che gli esseri umani ne hanno. Intuitivamente, gli input avversari sembrano totalmente diversi dagli input originali agli occhi dei modelli di Deep Learning pur essendo molto simili per gli esseri umani.

Fin dalla prima scoperta degli attacchi avversari contro le reti neurali, sono state proposte numerose contromisure e tecniche di rilevamento, ma difese affidabili contro questi attacchi sono tutt'ora una sfida irrisolta.

In questa Tesi presentiamo Defense by Massive Inpainting, una nuova difesa dagli attacchi avversari per i classificatori di immagini basati su reti neurali che sfrutta una totale assenza di fiducia dell'input. In particolare, la nostra difesa cancella completamente l'input e lo ricostruisce da zero grazie alla tecnica dell'inpainting.

Gli esperimenti sulla nostra difesa mostrano una maggiore precisione rispetto allo stato dell'arte contro la tipologia di attacchi più avanzata, vale a dire gli attacchi avversari white-box. Inoltre, riportiamo i risultati del confronto di tre diversi attacchi white-box su tre diverse reti rispetto alla baseline, scelta in quanto è l'unica difesa esistente che utilizza l'inpainting per correggere parti di input, seppur in modo molto diverso dal nostro.

Proponiamo anche un'evoluzione di tale baseline per consentire un'inferenza più rapida senza perdite di precisione, modificando il modo in cui essa seleziona le aree dell'input da correggere. Abbiamo chiamato questo secondo framework Saliency Based Localization and Inpainting. Tuttavia, questo nuovo modello non ha lo stesso successo di Defense by Massive Inpainting come robustezza agli attacchi avversari e non segue neanche un rigoroso approccio zero-trust, in quanto segue l'approccio della baseline.

Questo risultato conferma che la robustezza di Defense by Massive Inpainting è dovuta al suo approccio zero-trust, che non prende per vera alcuna informazione incorporata nell'input avversario senza prima purificarlo.

1 INTRODUCTION

Deep Learning based image classification models are becoming each day more present in our lives in order to help us in creating intelligent work applications or simply in order to replace the human need to perform many repetitive tasks.

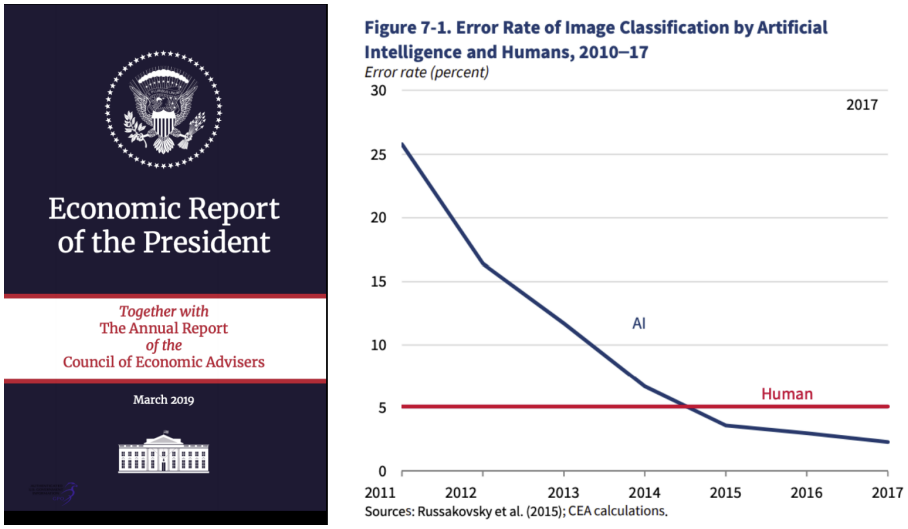


Figure 1.1: A silde from the 2017 Economic Report of the President of the United States of America talking about computer vision implications

Their increased usage and popularity is due to their high accuracy, such that the application of Neural Networks to image classification was even presented in the 2017 Economic Report of the President of the United States of America (Figure 1.1). Indeed, recent networks are now capable of outperform humans' brain in many object classification tasks.

Despite their good generalization, Deep Neural Networks suffer of a drop of accuracy in presence of small amounts of artificial generated noise in the image, namely adversarial perturbations. Thus, a lot of efforts have been spent in these years (Figure 1.2) to understand the deep functioning of this vulnerability present in Neural Networks models, how to exploit it to perform adversarial attacks, and how to defend from them. New defense methods are coming out on a daily basis and the research field is very active.

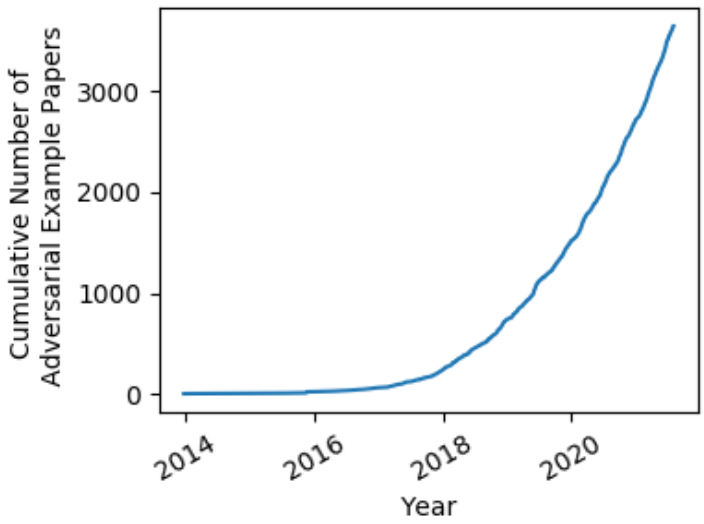


Figure 1.2: Cumulative number of papers about adversarial attacks since 2014 [63]

1.1 PROBLEM DESCRIPTION

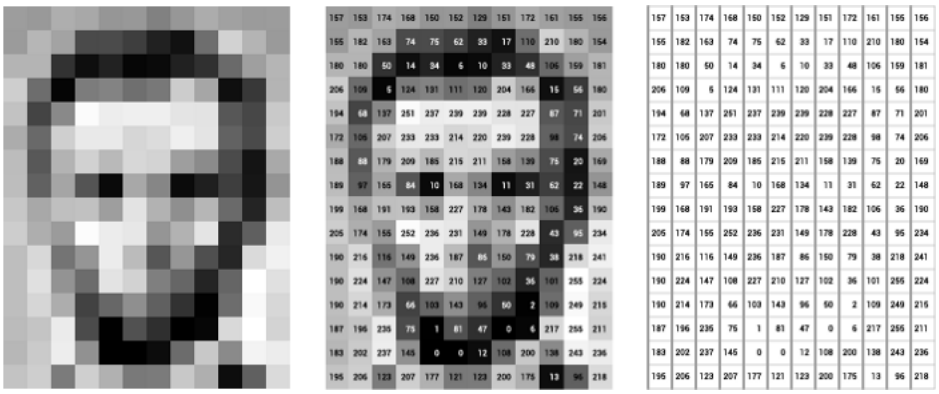


Figure 1.3: From left to right: a greyscale image of Lincoln with a resolution of 16px × 12px, the same image with the pixels labeled with numbers from 0 to 255 which indicate their intensity, and these numbers by themselves in a matrix form. Source: [61]

While there are only few comprehensive theories of our brain functioning on image recognition, a Neural Network (which is supposed to mimic our brain's computations) classifies images by only looking at the numbers which compose it, that are the pixel values. From a network perspective, it would be completely useless to classify the image of Lincoln shown at the left in Figure 1.3 because the classifier would need the pixel values shown at the

right. On the contrary, it would be hard for a human to guess what is shown in a picture by only looking at the rightmost matrix shown in Figure 1.3.

From a human perspective, the capability to classify an image based on our vision perception and not on pixel values makes us immune from these slightly, human-invisible changes, which objective is to alter pixel values in order to generate a misprediction of an image in a Neural Network.

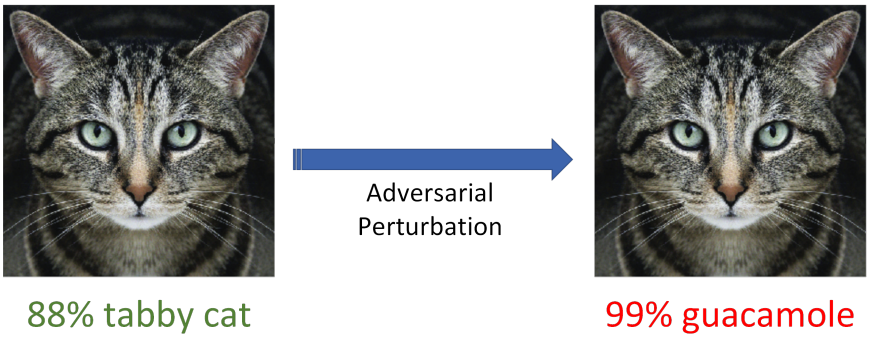


Figure 1.4: Two images of a tabby cat. The rightmost image is perturbed by some adversarial noise and the network mispredict it by labeling it as guacamole.

Consider Figure 1.4, the rightmost image is quite identical to the leftmost except a few pixels which slightly differs from the original. This small perturbation, if carefully executed, makes any Neural Network to mispredict the object shown in the image. In this case, the given model would perceive it as guacamole with 99% of confidence.

Unfortunately, the problem is far beyond the simple perception of the tabby cat as guacamole, because the same concept of slightly perturbation can for example be applied to physical object to fool the computer vision mechanism present in autonomous cars or any other computer vision system which process and classify images in order to take decisions on it.

At the center of Figure 1.5 it is possible to see a graffited stop sign, quite usual for anyone driving a car. Being a common situation, also self-driving vehicles should be aware of it and indeed the center image is correctly classified as a stop sign. However, if we consider the two other images, these are adversarially perturbed images that fake themselves to be a 45mph speed limit sign, and a self driving car which see one of the two mentioned signs, would accelerate instead of stopping.

It is clear that in safety-critical environments, we can not accept that a computer vision system may fail. But also in non-safety critical environments adversarial attacks can play a role. Consider for example the 2020 Congressional Candidate Andrew Walz shown in Figure 1.6. This image was never been shot: it is a computer generated image generated by a process known as DeepFake Generation. An high-school student just created some of these

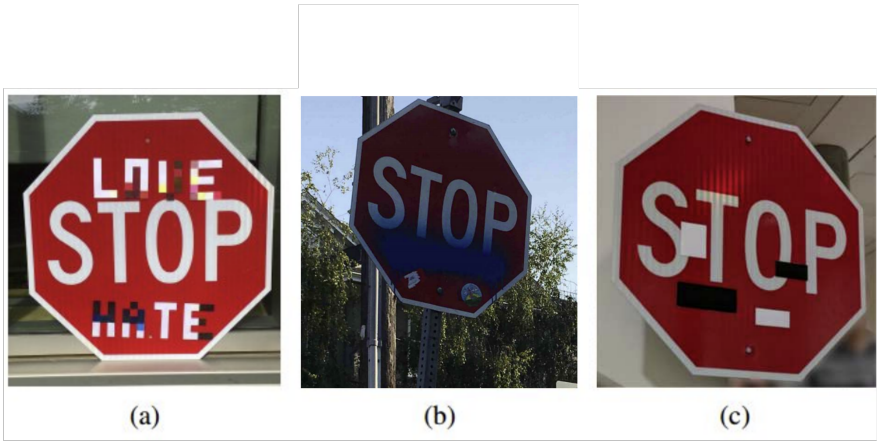


Figure 1.5: Three images of a physically graffited stop sign. At the center there is a real graffiti of a stop sign correctly perceived as a stop sign. On the left and on the right of it, we have a adversarial graffited stop sign which is perceived as a 45mph speed sign [58]

Figure 1.6: DeepFake image of a non-existent person who fooled Twitter into believing that he is a 2020 Congressional Candidate

images and a Twitter profile faking to be him, and he got the "Twitter Verified" blue check that was enforced by Twitter during the election period to avoid fake news and election manipulation. The motivation behind the blue check assigned to a fake person is related to adversarial attacks: nor Twitter Employees nor anti-deepfake systems were able to classify these images as fake, because such images contains a small adversarial perturbation, which successfully fooled the neural network DeepFake classifier.

Once again, it is not acceptable that all the neural networks performing computer vision can be fooled with such ease and scientists all around the world are struggling to find a way to defend from those attacks.

1.2 GENERATION OF ADVERSARIAL EXAMPLES

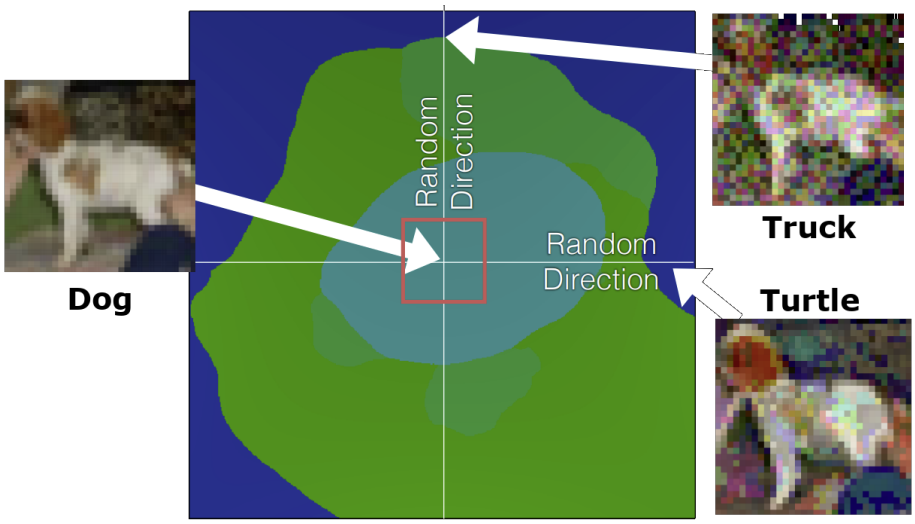


Figure 1.7: Visualization of 2 dimension in which we can perturb an image of a dog. The color represent the network confidence in recognizing a dog in the image. The light blue region is the area where the dog class gets more than 50% of confidence and thus in the light blue area the prediction is always correct. The green area instead includes a confidence in the true class lower than 50% but higher than 20% that means that the object is often classified correctly depending on other classes. The blue area means a confidence in the true class between 20% and 1%, while the yellow area (not shown here) would represent the area where the confidence of the dog class is lower than 1%.

But how are Adversarial Examples generated? Let's consider the 32px $\times$ 32px image of a dog shown in Figure 1.7 taken from [62]. In this figure, two random perturbation directions are shown. We can see that increasing the perturbation decreases the confidence that the network has in predicting the true class, but the two perturbed images on the right (misclassified as truck and turtle respectively) can not yet be called adversarial examples because the added perturbation is too high and it is difficult also for a human to identify the dog in those images. The boundaries shown in the figure are sufficiently broad to avoid adversarial attacks which follows these two directions. However the possible directions are $32 \times 32 \times 3 = 3072$, and we only picked two of them randomly.

In Figure 1.8 it can be seen that by carefully choosing the way in which we want to perturb the image, it is possible to find adversarial directions in which the perturbation needed to fool the classifier is minimal, and thus follow that directions. In this case the image classified as airplane on the bottom right and the original image classified as dog on the left are quite

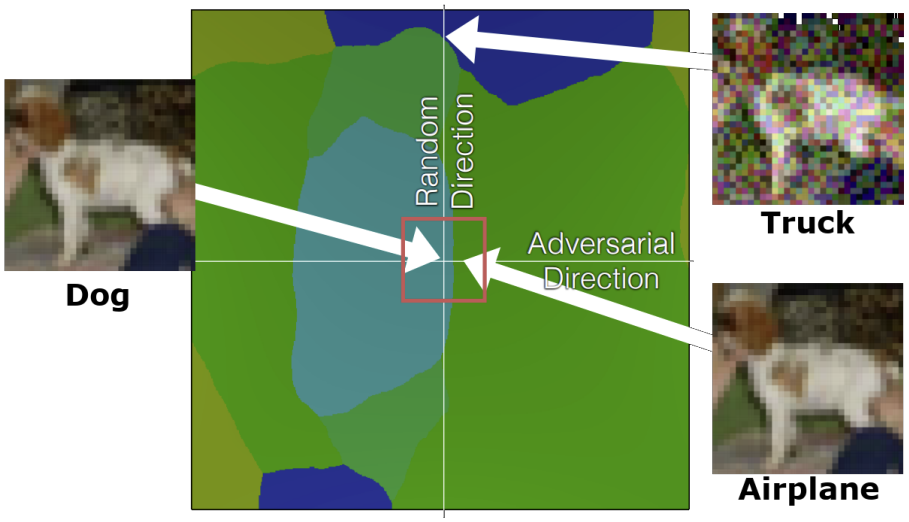


Figure 1.8: Visualization of 2 dimensions in which we can perturb an image of a dog, where one of them is an adversarial one. The color represent the network confidence in recognizing a dog in the image. The light blue region is the area where the dog class gets more than 50% of confidence and thus in the light blue area the prediction is always correct. The green area instead includes a confidence in the true class lower than 50% but higher than 20% that means that the object is often classified correctly depending on other classes. The blue area means a confidence in the true class between 20% and 1%, while the yellow area is where the confidence of the dog class is lower than 1%.

identical from a human point of view, but successfully manage to fool the network.

Figure 1.9 is the plot in three dimensions of Figure 1.7 and by looking at it we can see that the convex optimization problem of finding adversarial examples can be simply solved with traditional gradient descend algorithm, powerful and fast ways explicitly designed to follow the gradient direction until a local minima.

To avoid this, what scientist has done as first try was to embed layers which are not differentiable or highly irregular in order to mislead an adversary which want to follow the gradient direction to find the best adversary direction to perturbate the image.

The resulting confidence surface when a non differentiable layer is present is shown in Figure 1.10. Compared with the original surface shown in Figure 1.9, we see that having added to the classifier an irregular layer did not allow anymore attackers to exploit common gradient based method. However this entire class of defenses was proved to be ineffective as it is possible to build a substitute smooth model which imitates the boundaries of the original one and has the same high-level shape of the gradient, then attack such surrogate

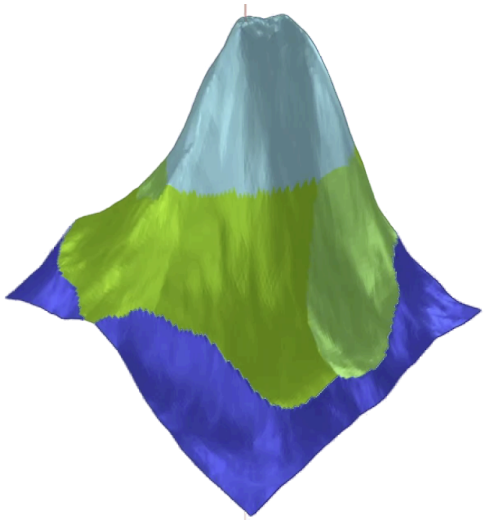


Figure 1.9: Plot in three dimensions of Figure 1.7.

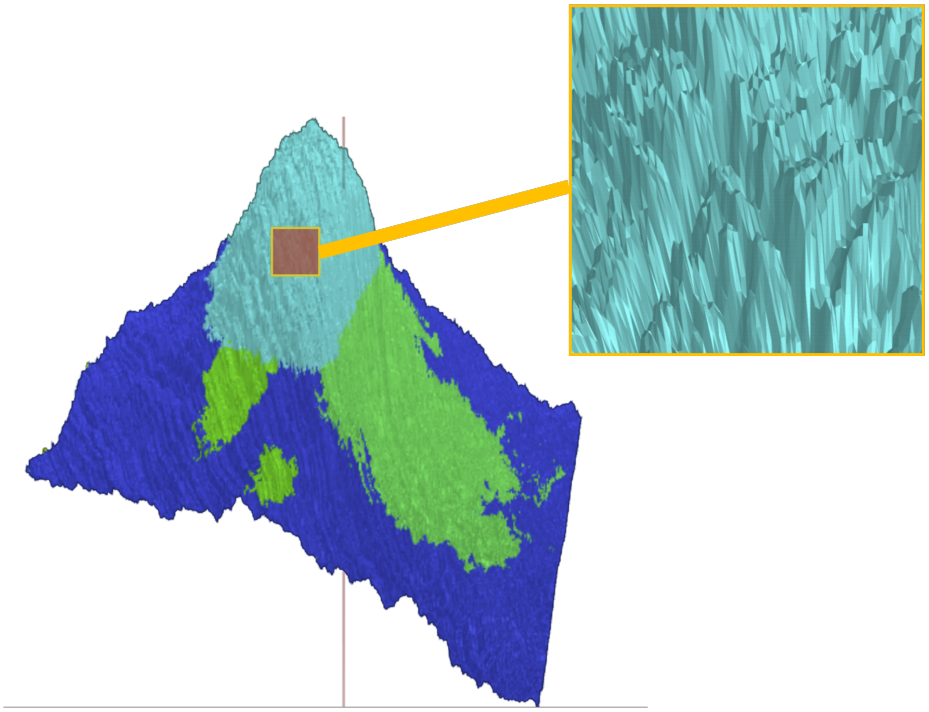


Figure 1.10: Plot in three dimensions of Figure 1.7 with a non differentiable, highly-irregular gradient.

model via common convex optimization methods and finally transfer the generated adversarial images back to the original classifier. In such a way

the generated images will fool the classifier with irregular gradient and the defense is ineffective.

In general, no effective and complete defense has been found so far, and in this Thesis we want to explore this field of research in order to find a new defense mechanism which can attenuate the problem of adversarial attacks.

1.3 METHODOLOGY AND DOCUMENT STRUCTURE

The structure of this Thesis will follow our progresses during the work. Each chapter represent an idea we had developed and is composed of an introduction focused on the theoretical aspects, hypotheses and motivations of our decisions, followed by a presentation of the experiments we did along with the comments associated with them. Eventually each chapter is concluded with a discussion of some key findings which made us proceed in the research and links to the subsequent chapter.

A summary of each chapter is as follows:

- **High level overview**: In Chapter 1 we present a general overview of the critical aspects from different points of view along with the scenario in which our Thesis was developed.
- **Problem formalization and state of the Art**: In Chapter 2 we provide to the reader the needed background to this work about the problem and the state-of-the-art both for attacks and defenses. We clarify the taxonomy of such field and we clearly state where our work collocates.
- **CIIDefence baseline**: In Chapter 3 we present and motivate the choice of CIIDefence as our baseline. We analyze in depth its functioning, spot areas of improvements and we pose the basis for our work to begin.
- **Saliency Based Localization and Inpainting**: Thanks to the knowledge acquired from our baseline, in Chapter 4 we develop some ideas to upgrade our baseline in order to improve both time and accuracy performance. In this chapter the main idea behind the results of this Thesis started to arise.
- **Defense by Massive Inpainting**: Our core idea that started in the previous chapter was developed in Chapter 5 and led us to the main result of this work. We present motivations about each choice and we identify the best hyperparameters via ablation studies.
- **Conclusion**: In Chapter 6 we comment all the journey of our work, integrating it with personal considerations. A brief summary of what we achieved is written there along with some possible starting points for future works.

1.4 CONTRIBUTIONS

In this Thesis we perform a research work around the field of Digital Adversarial Attacks on Deep Learning image classifiers. Our main contribution are:

- We analyze in-depth the work of Gupta and Rahtu about CIIDefence and we identify some critical aspects in it. We state a way to resolve them and we propose a possible explanation for the different performance that they have.
- We show an evolution of their work toward a zero-trust approach. However, in order to maintain a localization stage we need to stop before reaching a complete zero-trust strategy. We analyzed the performance of our new strategy and we reach great time improvements without any loss in the accuracy.
- We suggest a saliency based method instead of Class Activation Maps in order to purify areas of the adversarial examples before submitting them to the classifier to increase its efficiency.
- We propose a novel strategy, name lyrics Defense by Massive Inpainting, which make use of a full zero-trust approach applied to the input in order to be more robust to adversarial attacks. We stress each stage of our strategy in order to be methodological compliant with the state-of-the-art.
- We show how it is possible to employ a Generative Adversarial Network in order to transform all the adversarial perturbation into standard noise. We explain possible trade-offs between the level of noise generated and the level of adversarial perturbation removed.

The experimental results show that our proposed Defense by Massive Inpainting outperforms the CIIDefence baseline and is more resistant to full-white box attacks where the attacker knows, in addition to the classifier structure, also the defense mechanism in front of it. Compared with the state of the art our defense has similar performance and is fully compliant with the proposed methodological order proposed by [62].

2

BACKGROUND

Since the advent of convolutional neural networks for image classification, the computer vision field has seen many important milestones such as Deep Networks, Inception nets and ResNets, all of them aiming to further improve the classification accuracy on many image classification tasks. However, despite their high and still increasing accuracy, all those models were proved to be intrinsically weak to small perturbation of the input data called adversarial examples [73]. Even worst, some years later Kurakin, Goodfellow, and Bengio proved that while networks tend to be weak to perturbations, attacks do not (Figure 2.1). More formally, an adversarial example is an input data that has been intentionally perturbed very slightly in an artificial way intended to cause a neural network classifier to make a wrong prediction. These types of attacks can thus constitute a serious threat in many real systems because they may compromise the correct behaviour of deep learning models. For example, adversarial examples can be printed on standard paper and still be misclassified if captured trough camera app (see 2.1).

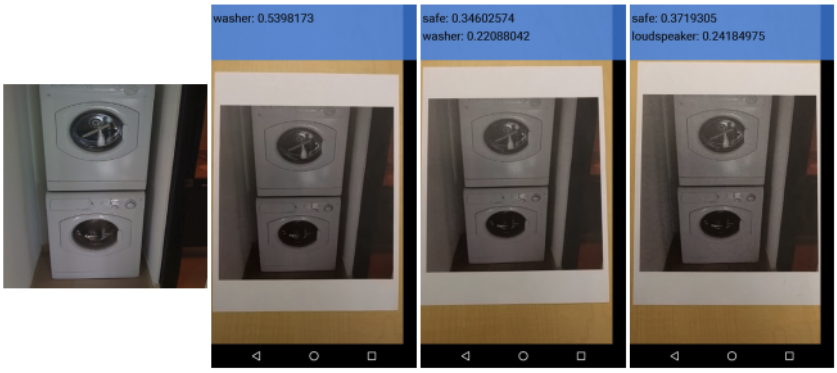


Figure 2.1: Demonstration of a black box attack performed on a phone app for classification in [20].

For instance an adversarial example crafted on a sticker and attached on a road sign might cause an autonomous car to perform different actions than it is intended to do by that road sign, causing a danger. Likewise, adversarial examples applied on human faces might fool face recognition systems, making those systems unreliable.

2.1 TAXONOMY

2.1.1 *Terminology*

As the field of adversarial attacks of a neural network is very wide, in this section we try to give an explanation of common acronyms that will be used later in this thesis.

- ILSVRC: Being the acronym of ImageNet Large Scale Visual Recognition Challenge [68], this competition takes place each year, where teams from all around the globe compete in different challenges such as image classification or image localization. The interest behind this competition has generated a lot of research going in the direction of achieving a performance boost in this challenge, often leading to new network architectures or new techniques.
- Adversarial Example and Original Input: Those two terms refers to the image submitted to the model for the prediction. While the Original Input is an image that has not been tampered, an Adversarial Example is an image artificially altered with imperceptible perturbations added to fool the classifier. Following the definition of Szegedy et al. in [48]:

  > «We expect networks to be robust to small perturbations of its input, because small perturbation cannot change the object category of an image. However, we find that applying an imperceptible non-random perturbation to a test image, it is possible to arbitrarily change the network's prediction. These perturbations are found by optimizing the input to maximize the prediction error. We term the so perturbed examples *adversarial examples*.»

2.1.2 *Network architectures*

In the last twenty years a huge number of network architectures were proposed and many of those brought with them new ideas and techniques to improve their performance such as decreasing the time needed to train, increasing the speed in predicting a sample, or increasing the classification scores. While there are a lot of those architectures, some of them are widely used especially in comparing performance between different research works. Here we briefly list the ones that we will use in this thesis and will give a hint about their structure and behaviour, being conscious that this list does not want and can't be an exhaustive explanation for which the reader is invited to read the original papers describing each network.

- VGG16: Originally proposed by Simonyan and Zisserman in [69], VGG16 has achieved state-of-the-art performance at ILSVRC2014, where it scored 92.7% as top-5 accuracy, quickly becoming a must in research papers. Compared to previous winners of the competition, VGG is way deeper and it features convolution layers of size 3x3 along with a maxpool layer of size 2x2. The drawbacks of this network architectures are its extreme slowness in the training phase and the heavy weights pack it needs which size up to half a GB. In the end, VGG16 is still considered an excellent vision model and it is widely used to compare performance across various research projects, reason why we have chosen to benchmark our defense on it. Another motivation behind the choice of VGG is that we want to explore how much the deepness of a network influence its robustness to adversarial examples.

- ResNet-101: Being the winner of ILSVRC2015, the name of this network comes from the words "Residual Network" and it was originally presented by He et al. in [70]. After the success of VGG16, all the networks adopted very deep architectures. This process however caused the so-called vanishing gradient, a state where deep neural networks lose their gradient due to their deepness. Indeed, during backpropagation, the gradient of earlier layers is obtained by multiplying the gradients of later layers and if those gradients are less than one, their multiplication vanishes very fast. Residual Networks solved this problem by using skip level connections between layers to transmit information even in the case of vanishing gradient. In addition, Residual Networks include blocks starting and ending with 1x1 convolutions in order to reduce the number of parameters and the time complexity while not degrading the network performance too much. Other than being a state-of-the-art architecture, we will use ResNet in this thesis because we know that defeating adversarial attacks often starts from image restoration but those techniques cannot recover the same level of details of the original. We then want to investigate how the skip-level connections employed in residual block can help in the classification of an image that has been firstly altered and then restored.

- Inception v3: Proposed the same year of ResNet-101 by Szegedy et al. in [71], the goal of Inception v3 was to reduce as much as possible the network complexity that VGG16 had brought in the year before. To accomplish this challenge, Inception v3 uses three different block types, each one of them aiming at reducing the number of parameters in a different way. Thanks to factorizing convolutions and efficient grid size reductions, Inception v3 was the first runner up at ILSVRC2015 scoring comparable performance to the state-of-the-art but having very few parameters if compared to both VGG16 and ResNet-101. Being

complementary to VGG, we selected this model to understand possible correlation between the number of parameters and the adversarial robustness. In addition, vanilla Inception v3 operates on input size of *299x299* while the other two network require an input image of size 224x224. Investigating this architecture, then, may also reveal correlations between adversarial robustness and the image input size.

2.1.3 *Types of adversarial attacks*

In the last years, Machine Learning has become pervasive is our life. New systems and models are being deployed in every domain that can be imagined, leading to a widespread deployment of software based inference and decision making including but not limited to, neural networks. Such pervasiveness has generated a lot of attention in finding potential attacks, that now space towards many possible application fields. In this section we want to give to the reader a basic view on attack types, and then we will focus on the types of adversarial attack we will use in this thesis. In 2016, Papernot et al. in [38] proposed the division of Machine Learning attacks in two categories:

- **Attacks on confidentiality and privacy**: this category of attacks attempt to expose the model structure and parameters (which may be a valuable intellectual property) or the data used to train it. This latter class of attacks is particularly harmful as it specifically target the privacy of the data source, for example in a clinical context, it could compromise the privacy of patient clinical data used to train medical diagnostic models. Potential risks are due to the extraction of the training data starting from the model's predictions [40], to adversaries performing membership test to discover whether an individual data point is in the train dataset or not [39], or to the possibility of recovering partially known inputs using the model to complete an input vector [40].

- **Attacks on integrity and availability**: this category refers to attacks attempting to influence the model outputs. As illustrated in Figure 2.2, depending on which context the Machine Learning application is working, an attacker may have different goals such as mining the availability or breaking the integrity. In an availability perspective, consider a network intrusion detection system that shutdown the network infrastructure if an attack is detected: by crafting adversarial example in the form of network traffic, an attacker may fool the IDS making it believe an intrusion is in place and thus causing availability problems (the network is shut off) even without being able to breach the security of the network infrastructure.
  Attacks focused on integrity, are a slightly different variation of availability attacks in which the attacker's goal is to compromise the behaviour

of an agent by forcing it into doing actions different to those it should do. Both the attack surface and the attack methods are the same for both availability and integrity attacks, but in the latter we can imagine an autonomous driving vehicle mispredicting a stop sign and keeping going on without stopping, or an autonomous stock price trader going all in on a specific stock because it was fooled with small (adversarial) variations of its price, while in availability attack we just need the architecture to stop working.
It is important to note that the example considered are referring only to attacks performed in the testing phase. However, attacks can be executed also in the training phase, for example by injecting into the train dataset specific adversarial input in order to compromise the entire learning process and thus the integrity and/or availability of the network [24].

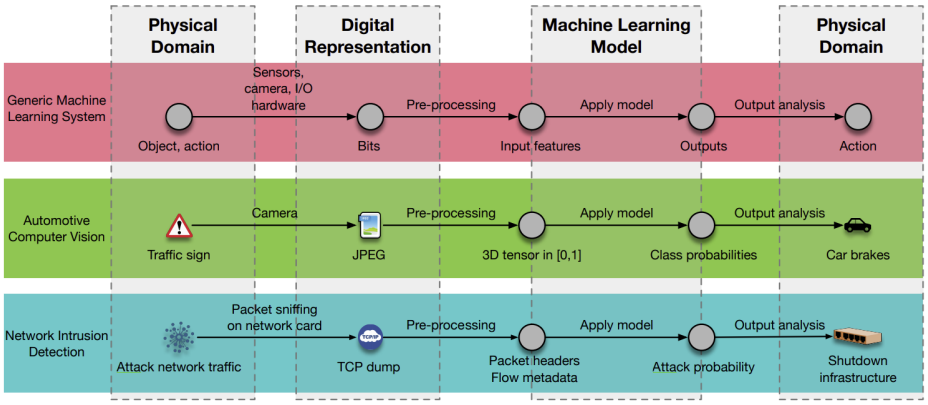


Figure 2.2: A good visualization of the general context in which an adversarial attack is inserted taken from [38]. The generic machine learning pipeline on the first row is illustrated with two examples coming from a computer vision model used by an autonomous driving architecture to interpret traffic signs and a network intrusion detection system.

For the rest of this thesis, we will focus only on attacks on integrity and availability. As anticipated, this category contains two different types of attacks with respect to the phase in which they are executed, that are:

- **Evasion attacks**: Performed in the testing phase, this attack assumes that the attacker has no way to influence the network different than altering its input. The training phase happens before the execution of the attack and there is no way for the attacker to influence it. The goal is to perturb the image as little as possible while causing a misprediction in the architecture.

- **Poisoning attacks**: Performed during the training phase, in this attack the adversary is able to inject custom data points in the training dataset or to directly perturb the images of the training dataset before they reach the network. This alteration of the training dataset affects the performance of the network, and the goal of this attack is to inject or to alter the lowest number of images in order to degrade the network accuracy on a clean test dataset as maximum as possible.

Being the evasion attack the most common and most studied case we will focus on this attack scenario, that can still be divided with respect to the type of the input that the adversary provides to the architecture, in these two categories:

- **False Positive**: this input is classified with high confidence by the classification model but is not recognized as belonging to any specific class by a human. An example of this attack is shown in 2.3
- **False Negative**: the classification of this input is obvious and straightforward for a human but the classification architecture fails into classifying it correctly. An example of this attack is shown in 2.5.

Figure 2.3: An example of a false positive adversarial example. This input can't be recognized by any human as belonging to any class, but is perfectly recognized with high confidence as a panda by a fooled classification model.

The most studied type of adversarial input is the false negative. Indeed, for this type of input attackers try to generate images that belong to different classes with respect to what an human see straightforwardly in them. In addition these images are often crafted by using as a starting point a real input and perturbing it with the minimum possible adversarial noise. By doing that, a human will encounter much more difficulties to spot the attack and the amount of perturbation added is minimized (in the false positive

input instead, if we start from a real input we would basically need to alter each pixel with high perturbations). Following the literature, we will focus only on false negative adversarial example generation attacks.

Still, evasion attacks on integrity finalized on the creation of false negatives can be divided in two groups based on the knowledge the attacker has on the model.

- In *white-box attacks*, the attacker has full knowledge about the classification architecture, that is, it is possible to exploit information related to the training data distribution, to the training algorithm, to the classification model and to the internal model parameters. In white-box attacks, the adversary usually searches for the most vulnerable areas of the feature space and, having access to the internal weights, crafts non-random perturbations that once added to clean images cause a misclassification of the input. An important mention that will be used later should be taken here: in a white-box settings, the attacker should also be aware of any defense that is in place [5] on the architecture. This lack of information often causes many research work to claim white-box performance when the attacker is not aware of the defense but in many of them, if the attacker knows about the defense, it would be obvious how to bypass it.

- In *black-box attacks*, the attacker only has the capability to query the classification architecture as an oracle, that is, it is possible to submit any input to the classifier and collect each output that the network produces. Using this kind of techniques usually involve the gathering of information about the training data distribution, the training of a surrogate model artificially created to imitate the classification bounds of the original network, attacking the surrogate with white-box techniques and transfer the attack on the original model.

Similarly to previous works [1, 3], we will evaluate our defense strategy against white-box attacks in order to have comparable results with the literature. In addition, accordingly to how Carlini et al. rephrase the Kerckhoffs' principle [17] in *On Evaluating Adversarial Robustness* [5], a defense strategy can be robustly evaluated only providing the complete knowledge of it to the adversary, as in a white-box scenario.

2.2 STATE OF THE ART

2.2.1 *Adversarial Attacks*

Following what has been done with the network architectures, in this section we want to illustrate the attacks that will be used later in this thesis, along with

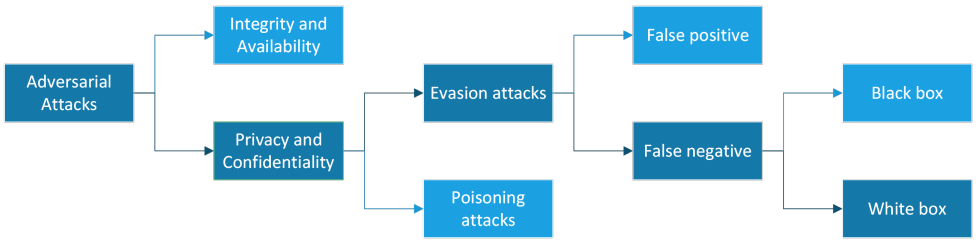


Figure 2.4: An overview of the relationships between the group of attack under study and the general attack division.

their functioning mechanism, their strengths and their weaknesses. As stated in the previous section, we believe that the most significant defense should be able to defeat white-box attacks, and so all the considered attacks will use the information available inside the model to craft adversarial examples.

- **Fast Gradient Sign Method**: FGSM is one of the first studied adversarial attacks. Proposed in 2015 by Goodfellow, Shlens, and Szegedy [73], it uses the sign of the gradient available inside the model pipeline to craft an adversarial example in a non-iterative fashion. As shown in 2.5, FGSM sums to the input image a small vector whose elements are equal to the sign of the elements of the gradient of the loss function with respect to the input rescaled by an imperceptibly small constant. Being x the input image, ε a small coefficient, ϑ the weights of the model, y the target associated with the input images and $J(\cdot)$ the loss function, the adversarial example is crafted as follows:

$$x_{adv} = x + \varepsilon \cdot \text{sign}(\nabla_x \cdot J(\vartheta, x, y))$$

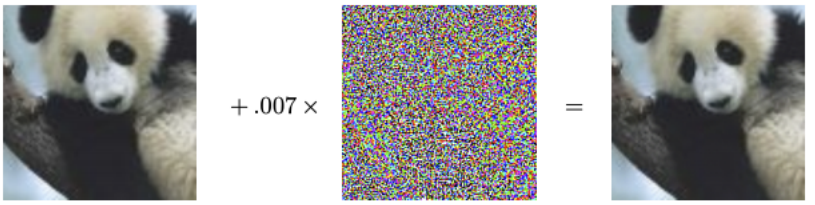


Figure 2.5: A demonstration taken from [73] of the Fast Gradient Sign Method applied to GoogLeNet (Szegedy et al. [49]) on ImageNet.

- **Projected Gradient Descent**: PGD is a well-know mathematical technique presented for the first time in [72] and its adaptation to neural networks is considered the most complete white-box attack because it reduces the effort the attacker needs to find the best attack by being quite always effective. The key point to understand PGD, is to consider it an attempt to find the perturbation that maximises the loss that a model has on a specific input while keeping the size of the perturbation - normally expressed as a L_2 or L_∞ norm - smaller than a specified amount, as shown in 2.6. To accomplish to this, PGD initializes the adversarial example randomly in the feasible set around the input image, and then loop until convergence the following steps:
  1. Take gradient step in the direction of maximum loss
  2. Project the perturbation back into the feasible ball

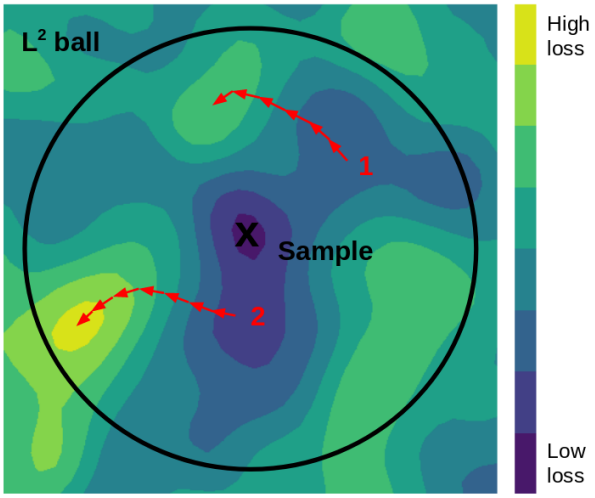


Figure 2.6: A demonstration taken from [60] of the Projected Gradient Descend method applied to generate an adversarial example which maximizes the prediction loss in the L_2 feasible ball.

- **DeepFool**: Despite not widely known as PGD and FGSM, this method is gaining a lot of attention thanks to its time-efficient adversarial examples generation and its smaller perturbations. Presented in 2016 [51], at each iteration DeepFool approximates the distance between the actual adversarial image x_{adv} and the boundary which defines the region of the space where the input x is correctly classified. Then, it computes the perturbation vector that reaches this boundary and updates the current estimate. These greedy iterations continue until x_{adv} is misclassified generating an adversarial example that often has

a smaller amount of adversarial perturbations with respect to other methods, as shown in 2.7.

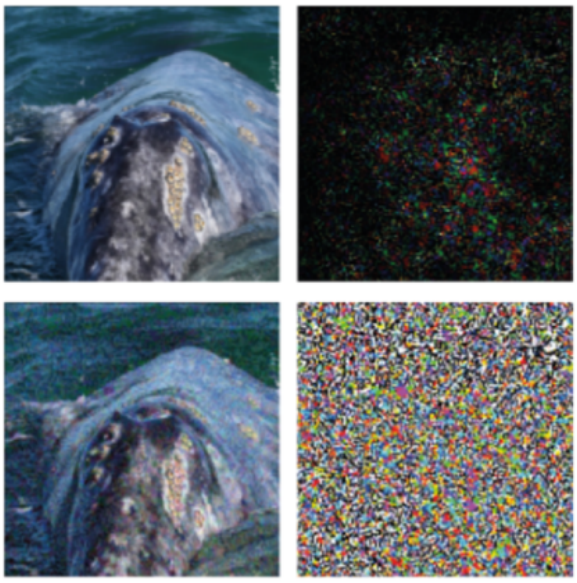


Figure 2.7: Comparison between DeepFool and FGSM taken from the original DeepFool paper [51]. In the first row it is shown the adversarial example generated by DeepFool x_{DF} classified as turtle instead of whale, along with its perturbation. In the second row instead it is shown the adversarial example generated by FGSM x_{FGSM} still misclassified as turtle, along with its corresponding perturbation. In this case, DeepFool leads to a smaller perturbation.

2.2.2 *Defense Strategies*

Considering the importance of the correct functioning of Convolutional Neural Network models in untrusted environments, many defense strategies has been developed. Most of those, however, are primarily focused on datasets containing a small number of classes or dataset containing small images. Even if these works are helpful in the research process against those attacks, such methods are not able to scale to large datasets like ImageNet. In addition, as defense methods should be as less intrusive as possible, Papernot et al. have identified four distinct characteristics that make a defense strategy preferable. Following their work in [4], a defense method should:

- **Have minimal impact on the architecture**: unless tested and proved, defense techniques should limit the modifications made to the architecture because introducing new architectures not studied in the literature requires a full analysis of their behaviour along with a careful design and benchmarking.

- **Maintain the original accuracy**: defenses against adversarial attack should not significantly decrease the model's classification accuracy on clean inputs.
- **Maintain speed of network**: the proposed solutions should not significantly impact the time that the classifier needs at test time. Indeed, the time consumed at test time matters for the usability of the model, whereas an impact on training time is acceptable because it can be viewed as a fixed cost.
- **Work for adversarial samples relatively close to points in the training dataset**: following the definition of adversarial examples, those inputs that are extremely far away from the training dataset are irrelevant to security because they can easily be detected, at least by humans.

In general, we can divide the defense strategies in five different non-exclusive categories, so that a defense can belong to many of them at the same time.

1. **Gradient Hiding**: provided the fact that we are considering white-box attacks where the attacker has full access to the model backbone and that most adversarial attack methods use the gradient of the model to generate adversarial examples, one can challenge himself if we have any possibility to deny the access to the gradient to the attacker. And indeed we have many way of hiding the gradient: consider for example a k-Nearest-Neighbor classifier that does not have any gradient at all. However, Papernot et al. in [2] shown that those models can still be fooled by constructing a smooth substitute model that imitates the boundary of the original one. Then, adversarial examples can be generated against it and then "transferred" to the initial model. This phenomena is known as *transferability of attacks* and makes any defense based on gradient hiding ineffective.

   Still, many modern models include a non-differentiable layer that prevent the attacker from obtaining the full gradient using the backpropagation algorithm. This technique, known as gradient masking has been proved ineffective in *Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples* [19], where Athalye, Carlini, and Wagner show how those non-differentiable layers can be easily approximated as linear to un-mask the gradient, proving also in this case the ineffectiveness of this defense.

2. **Robust training**: One of the possible strategies to create a more robust network at train time is based on adversarial training: introduced for the first time in [73], this defense increase the training dataset at runtime by adding adversarial generated examples. This is nowadays one of the most used methods for training robust models as the learnt network

representations empirically align better with salient data characteristics and human perception. In addition, as claimed in [21], even if robust models are less accurate on the dataset used to train them on compared to standard models, they often outperform the latter if measured on a transfer learning task. However, it should be noted that creating a robust model requires a full training of the architecture, and on huge datasets like ImageNet it is a heavy task. This is the reason why in the literature there are many papers on robust training on small datasets like CIFAR-10 [22], but very few papers analyze the robustness on big datasets. Another widely used method that falls in this category is defensive distillation which was originally presented in [74] as a compression method. Its adaptation to adversarial robustness consists of retraining a network using previously generated soft-labels from another network.

Yet even these specialized algorithms can easily be broken by giving more computational firepower to the attacker or by performing adversarial examples generation on a substitute model, as those techniques accidentally perform a sort of gradient masking [65].

3. **Randomization**: The idea behind this defense is to introduce randomness to the behavior of neural networks. Randomness-based defence methods add randomness to either the model parameters such as the weights or the gradient or they add randomness to the input image such as random resizing and random padding. As Wang et al. claim in [47], "in a deterministic AI model, attackers search until they find an adversarial example that fools that particular neural network. But if you have a random model, the attacker will need to find a better attack that can work on all the random variations of the model."

4. **Model ensemble**: Introduced for the first time in [44] and strictly linked with the randomness-based defense, ensemble methods can be used also for defence purposes. Besides just aggregating outputs from each model in the ensemble, some methods average the predictions over random noises injected to each of the model. In alternative, there can be introduced also a regularizer to incentive the diversity among the predictions of different models [46].

5. **Transformation based defences**: this wide category includes both input transformation and feature transformation methods. While in the feature transformation defences as feature squeezing, total variance minimization or image quilting, it is often applied a denoiser in the feature space, input transformation methods take advantage of the fact that several defenses aim at transforming the inputs right before feeding them to the classifier. Those techniques aim at reversing the

modification did to the adversarial images in many ways such as JPG compression [45] and wavelet denoising [23, 42, 41].

2.2.3 *Inpainting based defenses*

Special attention in this thesis will be placed on the concept of inpainting-based defense strategies. Inpainting is an important task in computer vision and refers to the process associated to the generation of missing or damaged pixels of an image in order to recreate an output that is the closest to the original input, before the pixels were cancelled or damaged. Nowadays, inpainting based techniques are widely used to remove small defects in photographs, to create images at a higher resolution than the original [18] or to restore corrupted images and artworks.

In recent times however, researchers have started to explore how inpainting based techniques could recover pixels from adversarial attacks as in "CIIDefence: Defeating Adversarial Attacks by Fusing Class-Specific Image Inpainting and Image Denoising" [1], or could help in discriminate if an image is adversarial or not [28]. The problem of image inpainting to restore adversarial pixels to the original state is not only related to the capability of the inpainting network to achieve the task but relies also on those architecture having something to start recovering from. Indeed, while adversarial attacks could be everywhere in the image, even the most advanced inpainting techniques require 75% of the image to be visible in order to recover the missing 25% of missing pixels [66]. Many techniques were proposed in the past to identify which areas of the image are more critical to inpaint than others and one interesting approach is the one proposed by Gupta and Rahtu which looks at Class Activation Maps of the top-5 predicted classes. We will go in depth of this technique in Chapter 3 and then will evolve the pipeline to a different methodology in Chapter 4 and propose a different approach in Chapter 5.

2.3 PROBLEM FORMULATION

As discussed in 2.1.3, we will focus on attacks on integrity and availability, in particular we will analyze attacks on image classification models. For those architectures, the problem can be formalized in the following way. Given a dataset $D : \{(x_i, y_i)\}_{i=1}^{N}$ where x_i is the i-th input to a machine learning classifier M and y_i is the desired output associated to x_i by M, an adversarial example associated to the i-th input is a different input $\tilde{x_i} = x_i + \varepsilon$ generated by imperceptibly perturbing x_i by ε such that the prediction that M outputs when $\tilde{x_i}$ is the input is $\tilde{y_i} \neq y$.

To make the perturbation added to x_i imperceptible, ε is bounded by either using L_0, L_2 or L_∞ norm. In this Thesis we will focus on attacks that use the L_∞ norm as constraint for ε.

Then, two categories of adversarial attacks arise from the last inequality:

- Attacks where the attacker force $\tilde{y}_i$ to a known output, namely *targeted attacks*. The adversarial examples generated by these attacks follow the gradient in the direction of the imposed output class $\tilde{y}_i$.
- Attacks where the attacker just impose the misclassification $\tilde{y}_i \neq y$: named *untargeted attacks*, this widely used category of attacks can minimize the adversarial noise ε better than targeted attacks because they just need to follow the gradient the the direction of the nearest classification bound.

As our goal is to maximize the accuracy under adversarial attacks, we will stick to the latter method that guarantees lowest perturbation while still influencing the classification accuracy.

2.4 RESEARCH DIRECTION

Considering the state of the art reported in this chapter about adversarial attacks and its possible defenses, it can be noticed the huge interest that this problem is having in the last years. In addition, considering the number of research works we can notice areas of big interest such as robust training, randomization and model ensemble and areas less explored such as transformation based defences. In the latter, very few attention is placed on inpainting techniques because of their drawbacks briefly discussed in 2.2.3. However, as the article "CIIDefence: Defeating Adversarial Attacks by Fusing Class-Specific Image Inpainting and Image Denoising" confirms, this defense methods can still be helpful and we will start from here.

We will run experiments to stress the pipeline proposed in [1] and then we will propose a different way to localize areas of interest adopting a deeply-supervised technique. We will progressively move from applying inpainting to small localized area to bigger ones. Finally, we will totally drop the localization step to focus on massive inpainting, proposing a novel defense called Defense by Massive Inpainting that will be discussed in Chapter 5.

3

BASELINE MODEL

We started our study by analyzing the current state of the art, as better detailed in Chapter 2. In doing this we found out CIIDefence, a novel technique proposed by Gupta and Rahtu in [1] combining both inpainting and weakly supervised localization as a defense strategy against adversarial attacks. As performing inpainting against adversarial examples has not been extensively explored in literature, we have chosen CIIDefence as our baseline model because it is one of the few proposing inpainting as part of the defense strategy. That is, we want to investigate the functioning of this methodology and possibly enhance its performance or to propose a new defense technique that incorporates some of the knowledge acquired from CIIDefence to get better scores.

3.1 DESCRIPTION AND METHODOLOGY

In this chapter we will detail in depth the functioning of CIIDefence in Section 3.1, but we still invite the reader that want to get even more information about it to read the original paper [1]. Then, as we believe that our results should be reproducible [5], we will give all the needed information about our setup and needed prerequisites in Section 3.2. Finally, in Section 3.3 we will describe and motivate our experiments while in Section 3.4 we will discuss the results obtained by the baseline and move to our original contribution. As our Thesis, also the CIIDefence technique is inspired by low number of recent works defending from adversarial examples with image reconstruction and denoising. In particular, while defenses like Pixel Deflection [3] - which they used as baseline - apply a sort of massive guided shuffling of all pixels in the input followed by a denoising filter, CIIDefence commits in finding in the adversarial image a few areas that are most influential to the classification result and altering those pixels that are present in those areas. To find them, Gupta and Rahtu had the idea to employ the Class Activation Maps as a sort of weakly supervised localization technique, able to localize in an image the pixels where the presence of a certain class is more probable. Then, they apply the inpainting method only for those small and carefully selected areas, that are the most relevant areas for each of the top-ranking class in the adversarial image prediction, and finally they use a non-linear wavelet denoising filter in the rest of the image to perform a sort of gradient hiding defense.

3.1.1 *Class Activation Maps*

As demonstrated by Zhou et al. in [6], Convolutional Neural Networks have the potential to perform a localization task even if those networks were trained only on classification based tasks. This idea is better articulated in [16] where Zhou et al. introduce for the first time a procedure to find the Class Activation Maps by selecting an arbitrary target class label and back-propagate the corresponding information throughout all the layers until the input image is reached. As shown in 3.1, each channel of the Convolutional

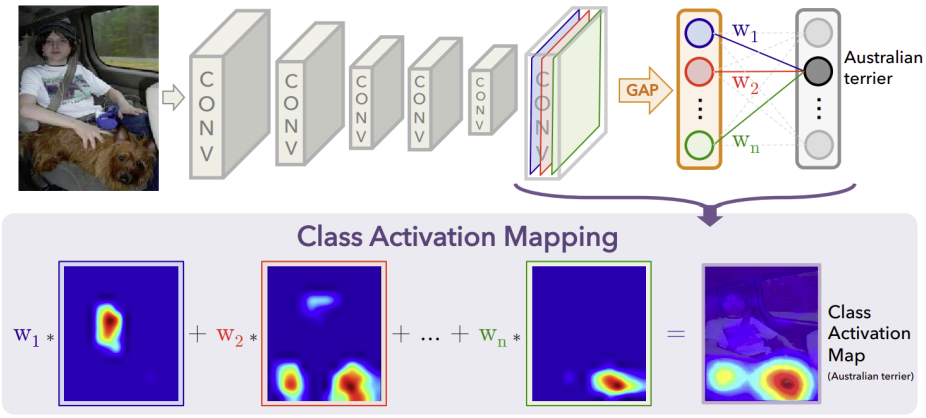


Figure 3.1: The predicted class prediction is back-propagated to the previous convolutional layer to generate the Class Activation Maps (CAM) at that layer to highlight the presence of some class-specific discriminative region. Image credit to [16].

Neural Network is activated by some patterns indicating the absence or the presence of a specific class in a certain area of the space. By indicating with $A^k(x, y)$ the presence of a specific class pattern in the k^{th} channel at the spatial location (x, y), the CAM M_c for a specific class label c is obtained as a sum of these responses weighted by the weights w_k^c of that specific class label c for the k^{th} unit.

$$M_c(x, y) = \sum_k w_k^c \cdot A^k(x, y)$$

It is important to notice that in order to use this method to get Class Activation Maps, the classifier need to end with a Global Average Pooling followed by the output neurons. In case of networks like ResNet there will be no problem, but if the model does not contain any Global Average Pooling layer it should be added to the architecture and the model should be retrained in order to identify the weights of the connection between the GAP layer and the output neurons.

3.1.2 *Inpainting*

Another method that we need to introduce before detailing CIIDefence is the inpainting. As said in Section 3.1.2, the inpainting task consists in recreating lost areas of the images in such a way that the inpainted image is the most possible similar to the original image. An example is shown in Figure 3.2.

Figure 3.2: An example of inpainting taken from [66]: on the left an input image with missing pixels, on the right the inpainted version of the image where those pixels were reconstructed.

Before the advent of Convolutional Neural Networks, researchers attempted to solve the problem using ideas similar to texture synthesis [7] where the missing regions are matched in other areas of the image and patched into the missing holes starting from low to high resolution. Such approaches are most suitable for background inpainting tasks but, as they assume that the lost areas can be found somewhere in the background, they cannot recreate totally lost image features for difficult cases where the regions to be inpainted involve non repetitive, complex structures such as faces or objects.

With the introduction of Generative Adversarial Networks by Goodfellow et al. in [37] and the rapid progresses in the field of deep Convolutional Neural Networks, many works started to formulate the inpainting task as a conditional image generation problem where high-level recognition and low-level pixel creation are jointed into a convolutional encoder-decoder network to stimulate the correspondence between recreated and already existing pixels [25, 33, 36]. However, even if these methodologies can synthesize plausible new contents, these CNN-based networks also happen to create blurry textures inconsistent with surrounding areas and those models also suffer of boundary artifacts near the inpainted regions.

In our Thesis the attention will be placed on a novel architecture called DeepFill v1 [66] and proposed by Yu et al. that tries to mitigate those problems. One of the motivation behind the widespread usage of DeepFill v1 is that it is one of the very few inpainting networks which have a public implementation

available. In general, DeepFill v1 consists in a unique Generative Adversarial Network which overcome to the described problems by means of two stages:

- **A simple dilated convolutional network trained with reconstruction loss to rough out the missing contents**. The novelty of this stage consists in the type of reconstruction loss they employed. In a masked image, the pixels that need to be inpainted located at the center of the hole have less similarities with visible pixel near the hole boundaries with respect to the ambiguity that have masked pixels near the border compared to their neighbor non-masked. This phenomena is similar to the reinforcement learning context when long-term rewards have big variations during sampling. In this latter case, the state of the art employs temporal discounted rewards, while in the GAN scenario the proposed loss is a spatial discounted reconstruction loss. The weight of each masked pixel is γ^l, where l is its distance to the nearest known pixel.
- **A contextual attention stage which makes use of the features of known patches as convolutional filters**. This stage consists mostly in convolutions to find the generated patches in the list of known contextual patches, a soft-max layer to weight differently each relevant patch and a deconvolution to recreate the newly-generated patches from the contextual patches. In addition, to allow the Generative Adversarial Network to create novel contents not present in the image, DeepFill v1 employs another path in parallel with the other which consists only of convolutions. The two paths are then merged together and provided to a single decoder architecture to obtain the final inpainted image.

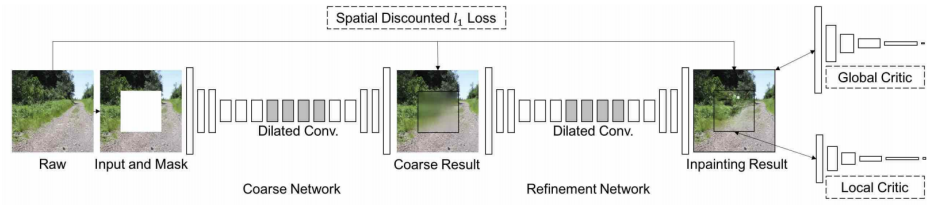


Figure 3.3: A general overview of the DeepFill v1 architecture taken from the original paper [66].

The general overview of the inpaitning architecture can be seen in Figure 3.3. As shown, another novelty of this network is the different losses that have been employed in the training phase. At early stages, it uses a pixel-wise reconstruction loss which, although tends to make the result blurry, is an essential ingredient for image inpainting. As described above, a spatial discounted loss is then applied to allow the hallucination of artifacts at

the center of the holes if the attention mechanism matches know patches near the boundaries. The refinement network which starts from the blurred inpainted image at the middle stage is trained using a modified version of the outperforming WGAN-GP loss. Indeed, this loss is not applied as is, but it is fused with the outputs of a global critic architecture and a local critic architecture that accounts for the discrepancy respectively between the general context of the image and the inpainted patches with respect to the image.

3.1.2.1 *DeepFill v2*

The inpainting mechanism explained above and used in our baseline comes from the work "Generative Image Inpainting with Contextual Attention" [66] and is named DeepFill v1. However in 2018, Yu et al. propose an evolution of such method named DeepFill v2. As in our experiments we will compare the results also using this upgraded version of the GAN we report here the functioning of this mechanism and the differences with DeepFill v1.

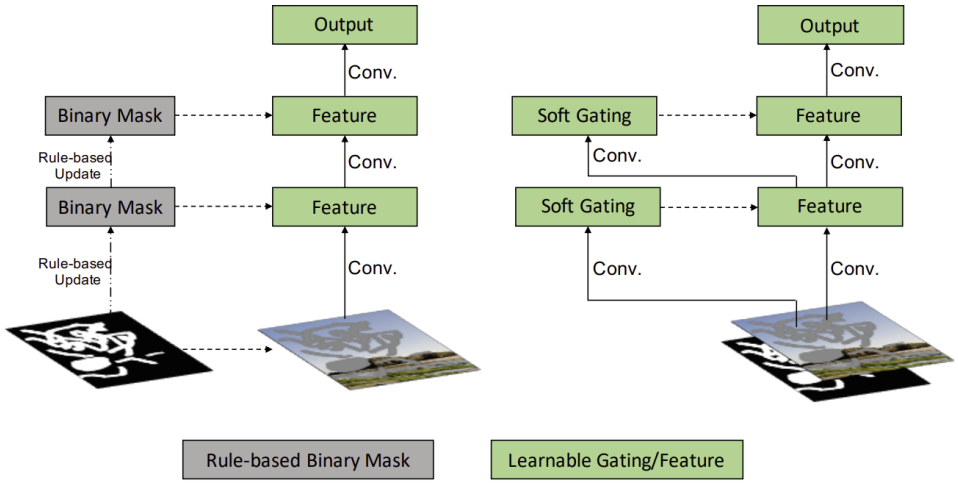


Figure 3.4: Illustration of the functioning of partial convolution guided by a binary mask (on the left) and gated convolution proposed in [66].

The main issue that Yu et al. wanted to address by proposing DeepFill v2 is that the approach of DeepFill v1 does not work for free-form masks. Indeed, if we think to train DeepFill v1 on irregular mask, its performance would not converge because of the behaviour of vanilla convolutions. This standard type of convolutions can only be applied on rectangular input shapes and if we would need to inpaint an irregular area they would perform convolution to the smallest rectangle containing the irregular mask, and not only on the masked pixels. To handle irregular masks in a correct way, Liu et al. proposed

the use of partial convolutions [30], a special type of masked convolution where the values are re-normalized to utilize valid pixels only.

As shown in Figure 3.4, partial convolution assigns to all the input pixels the value either valid or invalid, and multiplies a binary mask to those inputs throughout all the layers. However, even if this is an improvement, it has several problems. First of all, the input spatial locations across different layers may include valid pixels in input image but at a different depth also masked pixels in input image or synthesized pixels in deep layers. Statically forcing all locations to be either valid or invalid ignore these important information. Then, there are also problems related to user-guided image inpainting where users provide sketches inside the mask (are these pixels marked as valid or invalid?) and related to the lose of information that happens by using the binary mask at deeper layers.

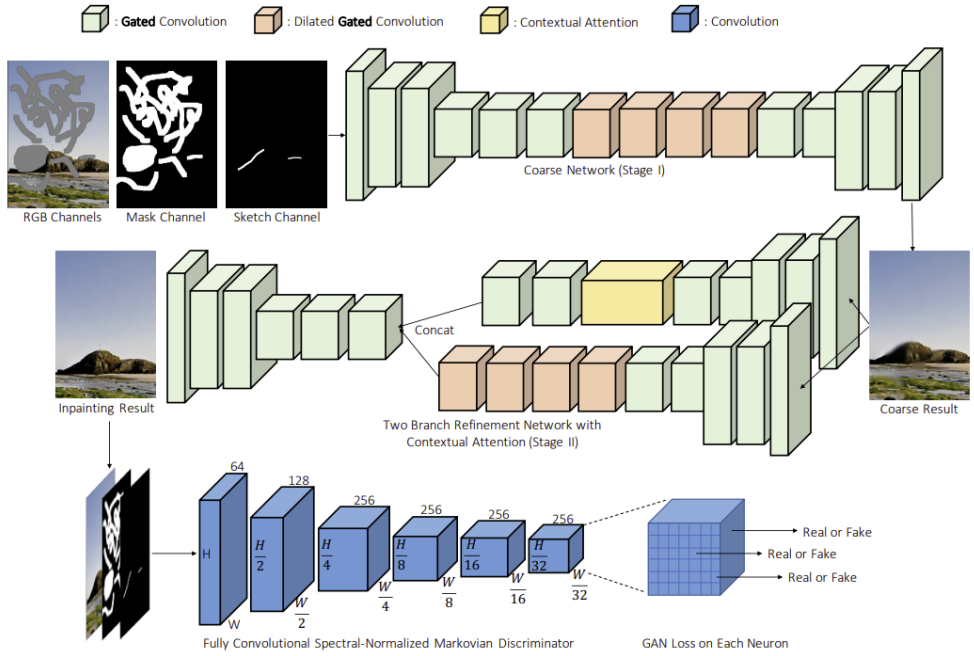


Figure 3.5: Overview of the proposed DeepFill v2 architecture taken from [67]. As it can be noted, after the inpainting we have the FCN whose objective is to discriminate which pixels were inpainted and which were not. Yu et al. apply their SN-PatchGAN in this stage.

Yu et al. proposed a different loss for their generative adversarial network, which they named SN-PatchGAN loss that applies on a fully convolutional network that act after the inpainting, as shown in Figure 3.5. The discriminator of SN-PatchGAN computes the hinge loss on each point of the output map with format $R^{h \times w \times c}$ and formulate $h \times w \times c$ number of different GANs

each focusing on a different spatial location and with a different semantics (with different channels depending on which is the input).

Another difference of this new architecture with respect to DeepFill v1 is that, given that free-form masks may be of any shape and anywhere in the image, global and local critic GANs created for a single rectangular mask such as DeepFill v1 would simply be not applicable and thus are not present.

3.1.3 *The functioning of the defense*

After having introduced the Class Activation Maps and the DeepFill v1 GAN, we can now better detail the functioning of CIIDefence. As said, this defense strategy aims at altering the adversarial examples in such a way that the key areas involved in the decision process would be perceived by the classification network as similar to those in the original clean image. CIIDefence mainly consists in three steps:

1. Image masking: In this stage the input image is submitted to the classification network in order to obtain the top-5 most probable class in the image. Then, based on the supposition that in those 5 classes there should be the true class, the image is processed by the same network used for the classification, with also the same weights, but modified at the end in order to produce in output the Class Activation Maps (see 3.1) for each of the top-5 classes obtained before. Once obtained those five CAMs, for each of them it needs to find the top 3 most influent areas for that specific CAM.
   In summary, for each of the top-5 predicted labels CIIDefence finds the top 3 regions of the space that are most relevant for the classification of those classes, for a total of $5 \cdot 3 = 15$ different areas (possibly overlapping). Then, a mask of the original image is created by leaving everything in place except those 15 areas, which are masked with a square size of 7x7.

2. Inpainting: In this stage CIIDefence submits the original image and the generated mask to the DeepFill v1 Generative Adversarial Network for image inpainting following the technique presented in Section 3.1.2. This network produces as output an image where all the non-masked pixel are exactly the same as the input, while the pixels that lie in those most influential areas for the classification have been inpainted and substituted. The key aspect here is that differently from many other works, in CIIDefence the key areas for classification are completely removed and then reconstructed on the basis of the contexts of neighbor pixels, and not just denoised starting from the adversarial pixels already present trying to remove the adversarial perturbations.

3. Image fusion: In the last stage of the defense, the original adversarial input is denoised with a wavelet-based filter and the key areas for classification identified at step 1 are replaced by their inpainted versions. In this way, the image is denoised everywhere except the key areas, which are inpainted. The motivation behind this choice relies in the observation they made that denoising the inpainted regions results in blurry areas which degrade the classification performance, while the motivation behind the use of wavelet filter among all the denoising filter is that they found out it to be not easily approximated by attacks which try to exploit gradient backpropagation to generate the adversarial example.

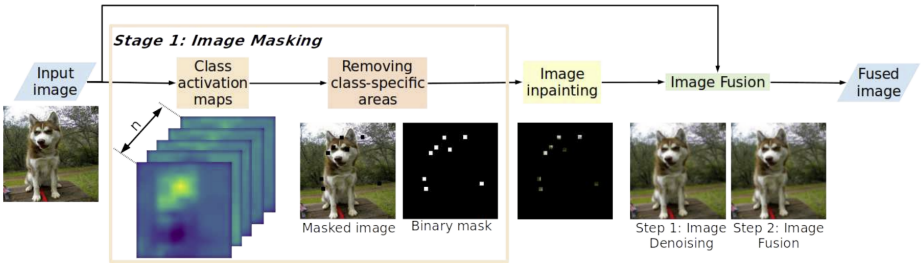


Figure 3.6: The workflow of the CIIDefence architecture taken from [1].

Once performed these three steps that hopefully purify the picture from some of the adversarial noise, the image is sent to the classifier model in order to get a prediction, as shown in Figure 3.6.

3.2 SETUP AND PREREQUISITES

Special attention should be placed on the setup and the prerequisites needed to run our baseline. Indeed, the original CIIDefence paper does not have any public implementation available. Given this lack of implementations, we have implemented it starting from the paper, with a few different details that should be noted:

1. GradCAM++ instead of standard CAM: We decided to test all of our experiments on the same networks used by Gupta and Rahtu, that are VGG16, ResNet-101 and Inception v3. However, as better specified in Section 3.1.1, in order to generate the CAMs for VGG16 and Inception v3 we would have to modify the original model and then to perform a full retrain on the whole ImageNet dataset, that consists of over 14 million images. Considering that in the original paper there is no mention about how they eventually performed this training, we decided

to use another CAM generation technique called GradCAM++ that was introduced in [35]. This technique is more advanced than standard CAM trough Global Average Pooling and does not need a retrain if the network does not have the Global Average Pooling layer. GradCAM++ involves weighting each k-th convolutional feature map A^k no more by the actual weights connecting the Global Average Pooling layer to the output neurons (it can be that no pooling is present) but instead of a quantity:

$$w_k^c = \sum_i \sum_j \alpha_{ij}^{kc} \cdot \text{relu}\left(\frac{\partial Y^c}{\partial A_{ij}^k}\right)$$

where y^c is the prediction for the class c and

$$\alpha_{ij}^{kc} = \begin{cases} \dfrac{1}{\sum_{lm} \dfrac{\partial y^c}{\partial A_{lm}^k}} & \text{if } \dfrac{\partial y^c}{\partial A_{ij}^k} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

2. The hyper-parameters used in CIIDefence for the wavelet denoising filters are not publicly available. We then decided to use the standard parameters offered by skimage which performs well and which are using BayesShrink as method, using multichannel processing and $\sigma = 0.01$.

In addition, some more consideration needs to be done on methodology differences between this Thesis and their paper.

- Like them, we only considered images that, without the presence of any attacks, were correctly classified. Investigating how a network behaves on images already misclassified even without adversarial examples would make no sense and this is the motivation why both us and CIIDefence have original accuracy = 100% on the ImageNet dataset, which consists of 1000 classes. However, they performed their experiments with just 3000 images randomly sampled from ImageNet dataset, which means on average 3 images for each class. Provided the fact that which subset of images they used is not known and that we believe that 3 images for each class are not enough, we have chosen for our experiments 10000 images, still randomly sampled. In addition, they evaluate the BPDA attack with just 500 images, that means that some classes are never gonna appear in the test set.

- When they proposed their work they used DeepFill v1, which used contextual attention. To avoid any loss of generality but at the same time experiment with a more powerful Generative Adversarial Network, we run each experiment twice: the first time we use DeepFill v1 as in

their work, the second time we use DeepFill v2 which makes use of gated convolutions and is a more advanced architecture.
However, they state that they used DeepFill v1 pre-trained on the ImageNet dataset. We believe that if we want to avoid any confusion, we need to ensure that no images in the test set were present in the train set of DeepFill v1, something that is just not possible to ensure as DeepFill v1 is pre-trained by Yu et al. on a subset of the ImageNet dataset which is not specified. In general, we believe that some (or all) the 3000 image they used could have been present in the Generative Adversarial Network train set. To avoid any objection, we used DeepFill (both version) trained on Places2 [34]. In such a way we also hope to verify the generalization ability that GANs have when performing the inpainting.

- Like them, we used the FoolBox library [26] to generate adversarial examples from clean inputs. However, they claim to have used adversarial images with L_2 equal to 0.03 but such value is too small to be used as L_2 norm on the ImageNet dataset. We will experiment with various epsilon values and for the comparison we will use an equivalent value of $\varepsilon = 2$ on L_{inf} attacks. With this parameter, we got images with an L_2 norm of the perturbation of approximately 500 which are comparable both in terms of adversarial perturbations and in terms of top-1 adversarial accuracy with their results. In our setting, each pixel (considering all the three channels) can thus vary of a maximum absolute magnitude of 12/255.

- Finally, according to [5] a defense mechanism like CIIDefence, which is based on a deterministic localization of the most influent areas, a deterministic inpainting of the those areas and a deterministic denoising of the background, can not claim white box performance without giving full knowledge of the defense to the attacker. In this case, an attacker with full knowledge of the defense would first perform the localization to identify the most influential areas and then it would add its perturbations in the area not covered by the inpainting. Indeed, the wavelet denoising filter can still be approximated by attacks designed to workaround non-differentiable layers, such as BPDA [19], thus CIIDefence is in our opinion weaker than described if operating in a full white-box scenario. We will take this into consideration throughout all our analysis.

3.3 EXPERIMENTS PERFORMED

In this section, we will describe which experiments we have run and what the motivation behind them. Then, we will continue with the results discussion in Section 3.4.

3.3.1 *Baseline assessment*

The first experiment we did is the assessment of our baseline. Indeed, considered the changes we have put in place and described in Section 3.2, we wanted to compare the performance of our architecture against the results reported in the official CIIDefence paper. Apart from the modification cited above, the experimental setting we used is the same as presented in [1].

3.3.2 *DeepFill v1 vs DeepFill v2*

By directly inspecting the generated masks we observed that it often happens that two or more squared holes referring to different categories overlaps as shown in Figure 3.7. This overlapping causes the holes not being perfectly squared anymore, as they can take any form that can generate from the overlapping of two squares. However, DeepFill v1 is not optimized to handle masks that does contain shapes of this kind. Indeed, quoting Yu et al. from the original paper of DeepFill v2 [67]:

> «Yu et al. [66] propose an end-to-end image inpainting model by adopting stacked generative networks to further ensure the color and texture consistence of generated regions with surroundings. *However, this approach is mainly trained on large rectangular masks and does not generalize well on free-form masks*. To better handle irregular masks, partial convolution is proposed where the convolution is masked and re-normalized to utilize valid pixels only.»

Starting from this observation, we have run a comparison between our baseline working with the first version of DeepFill and the same baseline working on the gated-convolution version, namely DeepFill v2.

3.3.3 *ImageNet vs Places2*

Recalling our second consideration in Section 3.2, we criticize the choice of running experiments on a test set composed of images taken from ImageNet and at the same time using DeepFill v1 trained on ImageNet's images. Yu et al. provide the GAN with a pre-trained model but they do not specify on which images the ImageNet version of DeepFill is trained on. This could

Figure 3.7: Example of a mask generated starting from an adversarial example of an ImageNet picture. It is possible to see the squares of different classes partially overlapping in the upper left corner.

cause situations in which some of the test images of CIIDefence could have been seen by DeepFill during the training phase. In addition we want to decouple the training set of the the GAN from the type of images we use as adversarial examples for testing purpose: this way we assess the ability of the pipeline to generalize to images of other domains. Considered those two points, we switch our baseline in using DeepFill trained on Places2 instead of ImageNet and still test the architecture using validation images from the ImageNet dataset.

In order to log all our changes, we decided to run a comparison between the performance of our baseline using DeepFill v1 trained on ImageNet as image inpainting technique and the performance of the same architecture using DeepFill v1 trained on Places2.

3.4 RESULTS DISCUSSION

We run our experiments on five different servers, each one of them equipped with a Nvidia K80 GPU and 12Gb of RAM, and an auxiliary server equipped with eight different Nvidia 1080Ti GPU each having 12 Gb of RAM. The results obtained are shown in the next sections.

3.4.1 *Baseline assessment*

Attack	Adv Top-1	Adv Top-5	Baseline ImageNet Top-1	Baseline Places DF1 Top-1	Baseline Places DF2 Top-1
Inception v3					
DeepFool	48.27%	88.32%	75.23%	75.87%	**76.01%**
FGSM	50.62%	83.73%	76.49%	**76.56%**	76.41%
PGD	35.78%	70.23%	70.02%	**70.22%**	69.70%
ResNet-101					
DeepFool	40.14%	85.18%	**64.06%**	63.50%	63.77%
FGSM	41.39%	83.33%	64.49%	64.26%	**64.51%**
PGD	30.59%	77.60%	58.41%	**58.72%**	58.17%
VGG16					
DeepFool	45.89%	86.88%	**55.58%**	**55.58%**	54.93%
FGSM	47.22%	84.80%	55.72%	**55.82%**	55.66%
PGD	42.67%	82.25%	52.74%	**53.09%**	52.44%

Table 3.1: Baseline results on the Inception v3, ResNet-101 and VGG16 classifier architectures with a small value of $\varepsilon = 0.2$.

In Table 3.1 we report the experimental result executed with $\varepsilon = 0.2$. In this setting, each pixels varies in a range that has an amplitude of 6/255 units. As the table confirms, PGD is the strongest attack between the three types of attacks we used (see 2.6). Consider that each attack should be executed three time to test each network as images that are adversarial for a specific network can be perfectly recognized by other architectures without any defense at all.

The first thing that the results show is that even using a small epsilon, adversarial attacks are able to make the network misclassify one image every two, dropping the top-1 accuracy with no defense to 50%. However, the adversarial top 5 accuracy measured without the defence is high and that is explained by the fact that the epsilon value is not sufficiently big to cause a decrease in the prediction confidence of the true class but the attack has anyway space to increase the probability of an adversarial class bringing it to the top, thus reducing the top-1 but not the top-5 accuracy.

It is possible to notice that our results present a drop of approximately 15% with respect to the results published in the official CIIDefence paper [1]. This can be explained both by the differences of our implementation (3.2) but also that we used a more significant subset of image that is more that 3

times larger than theirs. Indeed, it is possible to notice that the top-5 accuracy without no defense stabilizes around 85% thus setting an hard limit for a technique that relies and trust the fact that the adversarial top-5 accuracy contains the true class.

By increasing the value of ε up to $\varepsilon = 2$ (each pixel varies in a range of amplitude 12/255 units) we can observe in Table 3.2 the drop in accuracy that CIIDefence has, especially when attacked by PGD.

Attack	Adv Top-1	Adv Top-5	Baseline ImageNet Top-1	Baseline Places DF1 Top-1	Baseline Places DF2 Top-1
Inception v3					
DeepFool	34.96%	78.19%	**67.35%**	59.93%	54.30%
FGSM	35.55%	67.79%	53.81%	**53.94%**	53.56%
PGD	3.61%	11.86%	43.07%	**43.40%**	42.00%
ResNet-101					
DeepFool	15.15%	68.63%	**37.57%**	26.77%	26.31%
FGSM	15.33%	55.21%	31.18%	**31.30%**	31.00%
PGD	1.57%	30.33%	16.44%	**16.86%**	16.26%
VGG16					
DeepFool	14.29%	69.56%	**27.49%**	19.62%	19.34%
FGSM	12.55%	47.33%	**20.79%**	18.32%	18.20%
PGD	3.95%	36.14%	10.63%	**10.64%**	10.34%

Table 3.2: Baseline results on the Inception v3, ResNet-101 and VGG16 classifier architectures with an high value of $\varepsilon = 2$.

The results of the experiments executed with $\varepsilon = 2$ also rises attention on the difference between the three networks. Even if Inception and ResNet are comparable in terms of top-1 accuracy after the defense, there is a drop when we consider VGG16. Our explanation is that while VGG16 is a very deep convolutional neural network without any optimization for what concerns the high-level feature representation, ResNet and Inception are optimized to handle in their deep layers also fine grained features. In addition, those two networks also has fewer parameters (which increase the robustness in presence of adversarial attacks [27] even if it reduces the accuracy in a general setting) but at the same times are approximately 7x deeper than VGG16. Given that the inpainting technique adds a limited quantity of blurring and noise to the image, networks like Inception v3 with its asymmetric

convolution to promote high level feature representation and ResNet-101 with its skip-connection which maintains an high level of details, are less prone to suffer blurring and noise issues as much as VGG16.

3.4.2 *Considerations*

Those results set up our baseline target. However, there are a few thing to highlight:

- The test time of the CIIDefence architecture takes an extremely long time and this is mostly due to the CAM generation stage. This stage needs approximately 2 seconds for each image, which is a long time if we consider that this defense should be practical to use also in real-world scenarios. In our work, we will proceed to investigate if a more time-efficient way does exist.
- We executed experiments with small epsilon to evaluate the accuracy continuity across various epsilon ranges. The results show that, with such low perturbations, there is no difference in inpainting with a Generative Adversarial Network trained on ImageNet or Places2 and no difference in accuracy appear also with respect to the DeepFill version (contextual attention or gated convolutions). If compared with the results obtained testing the same baseline with an higher epsilon such as in Table 3.2, we can see that the GAN trained on ImageNet gives an extra +10% boost to the top-1 accuracy in case of localized attacks such as DeepFool. This would be explained by taking into account the considerations done in subsection 3.2 regarding the possible overlap between test images and GAN train images. In addition, this extra performance boost visible at higher epsilon values gives an hint on the fact that the loss on the accuracy at lower epsilon is mainly due to the technique and not on the specific inpainting method.
- Finally, we noticed that in all the experiments done on the baseline, the images that were correctly classified after the defense were all present in the adversarial top-5. That is, CIIDefence is able to recover a correct classification (top-1) from images that at least are in the adversarial top-5 accuracy measured without any defence. This is confirmed by the results, as no attack and no network architecture along with CIIDefence were able to generate a defended top-1 higher than the adversarial top-5, except for PGD executed with $\varepsilon = 2$ on Inception and ResNet as the denoiser stage is able to remove some of the perturbation added, if we consider that the attacked does not know about the denoiser structure (grey box scenario). Notice that Gupta and Rahtu in the CIIDefence paper does not tell which was the top-5 accuracy of the adversarial

images measured without the defense. Their *random extraction* of such a little number of images could have been a "lucky extraction" and can have selected images with an high top-5 and a low top-1. This is another element that can explain the differences in accuracy between our implementation and theirs.

This last point open up the door to an important reflection that will be the guideline for the entire thesis work. CIIDefence is taking the adversarial top-5 accuracy as a trusted zone, where it is possible to find useful information for the correct recovery of the true class. Indeed they are trusting these top-5 classes by computing on them the class activation maps and then inpainting the most influent areas.

Stimulated by the lowering of the top-5 accuracy as ε increases, we will investigate if it is possible to un-link our performance from the adversarial top-5 accuracy. In addition, we believe that given the high dependence between top-5 accuracy and final prediction, this defense exposes a new attack surface: an attacker could indeed exploit this functioning of the defense and modify its attack accordingly to kick off the true class from the top-5 most probable ones. We believe that this approach put too much trust in the adversarial image and we will try to switch to a zero-trust model in the following chapters.

4

SALIENCY-BASED LOCALIZATION AND INPAINTING

While we were working on the baseline, we soon realized that the CAM generation procedure was a true bottleneck for the time needed to run the experiments. A quick comparison between the standard CAM generation procedure on ResNet-101 (which does not require retraining) and GradCAM++ shows that the time needed for both methods is comparable, so it is not an issue of which version to adopt. To give an idea, the mean time needed by the architecture to generate the 5 different CAMs required for each image is around 2 seconds on a Nvidia 1080Ti GPU (12GB of Graphical Memory). Then the whole pipeline requires an additional second to inpaint, denoise and predict, for a total time of 3 seconds, 66% of which is needed only for the CAMs.

In addition to test time, we observed that by making attacks stronger (i.e. increasing ε), the top-5 accuracy started to drop really fast, making it difficult for CIIDefence to recover useful information by trusting the fact that in the adversarial top-5 classes it would find the true class. We thus developed a strategy less top-5-classes dependent, and more time-efficient, to generate a confidence region that guides the inpainting procedure.

Investigating this issue, we found out that while CIIDefence only employed DeepFill v1 which works better on squared masks, now we have the possibility to leverage the free-form inpainting of DeepFill v2 and thus we can create mask of any shape without caring about them being squared. The main idea of the defense presented in this chapter comes from "Deeply Supervised Salient Object Detection with Short Connections" [32], where Hou et al. propose a novel technique to identify salient objects in an image in less than 100 milliseconds, called Salient Object Detection. However, while in the CAM generation procedure we choose which class we want to generate the CAM, applying this technique only provide us a single mask where the salient object is highlighted but without no clue about its class, as shown in Figure 4.1.

By switching the guideline of the inpainting from the Class Activation Maps to a saliency-based approach, we not only avoid generating 5 different CAMs for each image (decreasing the time required), but we also take a step towards the direction of a zero-trust mechanism. As noted in Section 3.4.2, we can't trust too much information coming from an image for which the attacker is in full control, and we should try to have an independent defense strategy.

4.1 DESCRIPTION AND METHODOLOGY

Before replacing GradCAM++ with the Salient Object Detection method, we needed to verify that the network performing the Salient Object Detection task was invariant to adversarial attacks. That is, we have put in place experiments to verify if the Salient Object Detection network is still able to recognize the shape of the correct salient objects also in an adversarial example. Of course, the network would be unable to tell which class does the object belong to, but still it should highlight its edges. After having implemented this method by taking the source code and the pre-trained models from one of the unofficial implementations of the project [32] available on GitHub (link: [64]), we run an experiment which consisted in the classification of the salient object without no context nor background. Thanks to the results reported in Section 4.3.1.1, we proceeded in this direction and we integrated this method in our pipeline.

Figure 4.1: An example of the output of the Salient Object Detection technique taken from one [64] of the unofficial GitHub repositories referencing [32]. This technique does not need any input other than the image and creates this mask by means of short connections to skip-layer structures.

This new methodology does not provide us with a fixed number of pixels that are most influential to the classification (in case of CIIDefence it was $(7\text{x}7)\ \text{px} \cdot 3\ \frac{\text{times}}{\text{classes}} \cdot 5\ \text{classes} = 735\text{px}$), but it does provide instead a dynamic number of pixels depending on how big the salient object is with respect to the image.

However, as the saliency approach tend to output bigger masks with respect to the CAMs method, we modified the way in which we use the inpainting architecture following a *divide-et-impera* scheme: DeepFill will run on image with a mask size lower than a certain threshold and if the area to be inpainted in the mask is bigger than such threshold, the masked area is split

in smaller inpainting areas and inpainted independently, thus running the inpainting multiple times on different smaller masks. After this procedure, all the reconstructed areas are merged to rebuild the original shape identified by the salience mask. Even if we increased the number of inpainting areas to approximately 5, we did not suffer any performance issues in terms of timings as the DeepFill GAN architecture is really fast.

In this chapter we will show the evolution of this idea, starting from the original saliency-based approach with dynamic number of pixel (which is the approach proposed in [32]) and then switching to a modified version in which we limit the number of pixel to the N most relevant, where N is an hyperparameter.

4.1.1 *Salient Object Detection*

As many other strategies described in this thesis, also the salient object detection task has seen huge improvements since the introduction of Convolutional Neural Networks. However, pure CNN-based methods use Class Activation Maps to perform the localization tasks and thus their outputs are not fine grained, as shown in Figure 4.2. Indeed, as described before, CAMs and pure CNN-based methods are only able to provide a coarse weakly supervised localization but does not have the potential to output fine grained details.

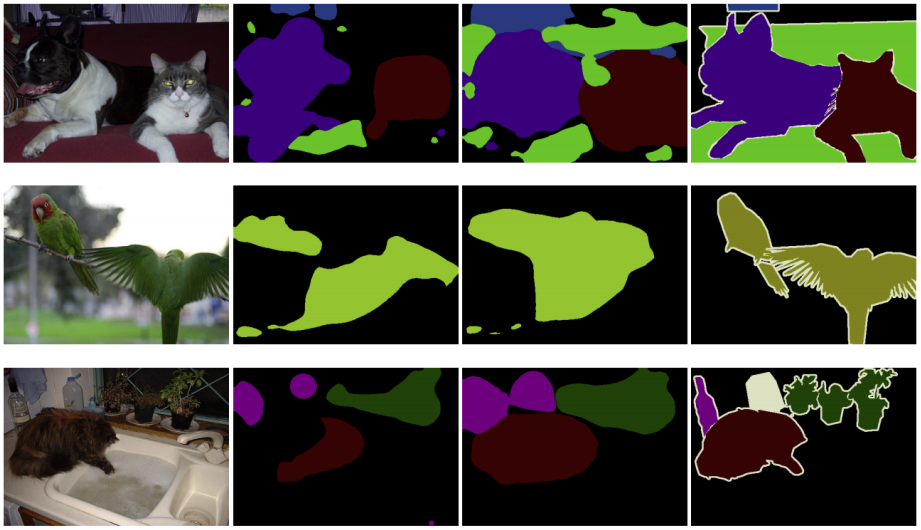


Figure 4.2: An example of the weakly localization possibilities offered using only pure convolutional layers taken from [29]. From left to right, the original image, two outputs of two different CNN-based localization methods which use convolutional layers, the ground truth.

As this is an hard limit for weakly supervised techniques, many research works proposed different workarounds to identify pixel-level details, such as [31]. Holistically-Nested Edge Detection (HED) is a method which combines the expressiveness of deep CNNs with nested multi-scale feature learning and provides in output directly the edge mask in an image-to-image fashion (you provide an image as input and the network outputs another image). It consists in a skip-layer structure (created by adding to each convolutional layer of a VGGNet architecture a side output) with deep supervision for edge and boundary detection that fuses multi-level features extracted from different scales in a natural way, combining information coming from the CNN such as the coarse class localization, with information coming from edge detection layers which make the boundaries fine-grained.

However, salience detection is an harder task with respect to the general localization just discussed since the latter does not rely too much on high-level semantic feature representations. In the Holistically-Nested Edge Detection, the output masks contains all the edges which are present, not just the most salients ones, in which we are interested.

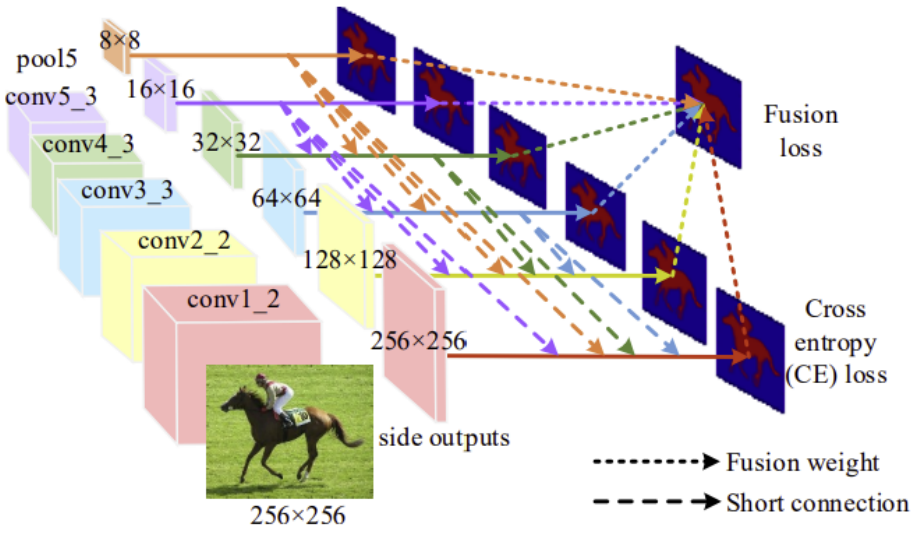


Figure 4.3: The salient object detection network architecture [32]. In the image the underlying network is based on VGGNet which has 6 different scales and thus introduces 6 side outputs, each of which is represented by different colors. In addition to the side loss for each side output, a fusion loss is employed to capture all the features coming from different levels.

Exploring the internal representation of HED networks, Hou et al. noticed that the deeper layers outputs encode the high-level semantic information about the salient objects and can localize them in these outputs. However, due to the down-sampling operations in FCNs, outputs of deeper layers

are normally small (at most a few pixels) and they have irregular shapes especially when the input image is complex. Shallower outputs instead can capture rich spatial information but are less meaningful if the goal is to identify how much the object is important in the final classification.

Based on these observation, the novelty in [32] for creating better saliency masks is to combine these multi-level features coming from deeper and shallower layers with the introduction of short connections to the skip-layer structure within the Holistically-Nested Edge Detection network.

Those connections from deeper layers to the shallower ones, offer two advantages:

1. High-level features from the deeper layers can help shallower layers to better locate the most salient region

2. Shallower layers can learn rich low level features to refine the sparse and irregular prediction maps from deeper layers

These short connections, linking the deeper side output layers to the shallower ones, can be mathematically formalized by writing the equation which computes the new side activation $\tilde{R}_{side}^{(m)}$ of the m-th side output using the weights r_i^m of the short connection from the side output i to the side output m (with $i > m$):

$$\tilde{R}_{side}^{(m)} = \begin{cases} \sum_{i=m+1}^{\hat{M}} r_i^m \tilde{R}_{side}^{(i)} + \tilde{A}_{side}^{(m)} & \text{for m=1,...,5} \\ \tilde{A}_{side}^{(m)} & \text{for m=6} \end{cases} \tag{4.1}$$

For what concerns the training of the proposed Salient Object Detection network, it is performed by means of deep supervision and followed the training procedure of state-of-the-art saliency detection architectures. In particular they claim to have used the same procedure as [59], which use 2500 randomly sampled images from the MSRA-B dataset [57], 500 images as validation set, and the remaining 2000 images of MSRA-B as test set.

MSRA-B [57] is a dataset which includes 5000 images each one containing labeled rectangles from 9 different users who were asked to draw a bounding box around what they consider the most salient object. This dataset feature a large variation among images which vary from natural scenes, to animals, cities, etc. For the saliency object detection training Jiang et al. manually segmented the identified salient objects edges within the user-drawn bounding box and thus obtained binary masks, which were used to train, validate and test the architecture.

In summary, by fusing information from different levels, the architecture benefits of rich multi-scale feature maps at each layer, an essential property to perform an efficient salient object detection.

4.1.2 *The functioning of the defense*

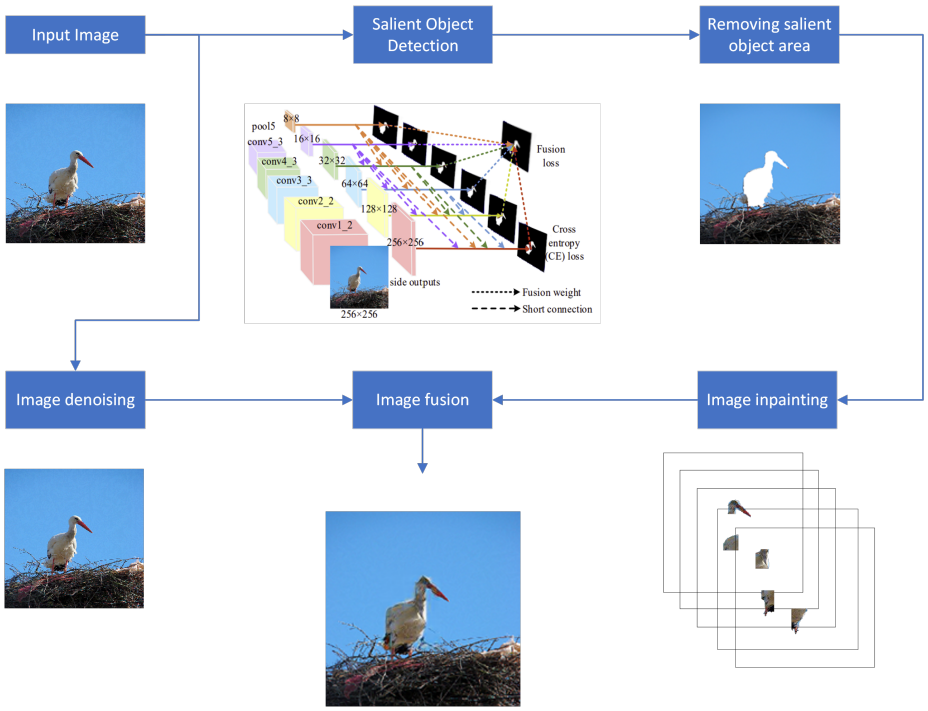


Figure 4.4: Our first trial using Salient Object Detection as a substitute for the CAMs localization in the CIIDefence pipeline.

We experimented the use of the original version of the Salient Object Detection mechanism, which returns a dynamic number of pixels depending on the object size, and used the same inpainting strategy as CIIDefence by reconstructing the pixels inside the salient area and denoising the ones outside that area, as shown in Figure 4.4. However, as the returned salient object size is dynamic, it can happen that it occupies up to 50% of the total image. As it is impossible for any inpainting network to hallucinate such a big content, to improve the inpainting performance we run the inpainting GAN following a *divide-et-impera* scheme which we named *Partitioned Inpainting*. In particular we run the GAN five times for each mask, each time inpainting 1/5 of the total mask and then fusing it all together (standard CIIDefence runs DeepFill only once as the inpainting area is constant). A detailed view of the inpainting result as described here is shown in Figure 4.5.

Observing the inpainted output it is possible to notice that the reconstructed pixels are more blurred with respect to CIIDefence, and this is due to the bigger inpaint mask size. Even in pictures where the salient object occupies an acceptable size of the total image, the inpainted outline is still not well

Figure 4.5: Comparison between the adversarial input and the image with the inpainted foreground after fusing the five different inpainted parts of the *divide-et-impera* scheme (partitioned inpainting).

defined as in clean images (which is an important feature for the classification outcome, as confirmed by CAM outputs). In images where the salient object is even bigger, some detail of it was lost.

To better explore all the details of this mechanism we also tried to switch the area to be inpainted from the foreground to the background, still using our partitioned inpainting. This choice is motivated by the fact that we do not know a-priori where the attack would focus more its adversarial perturbation, but for sure a clean and regular area such as the images background would be a suitable fit to place the perturbation. Switching from inpainting the salient object to inpainting the background has also a more deeper meaning: in average, we are reconstructing more than 50% of each image, and we are inpainting the areas which should be less meaningful for the classification outcome. In other words, we are decreasing the level of trust we have in the image. Even if the background would not be inpainted perfectly the classification network might not care too much: some blurry areas in the background should not change the classification of the foreground, but at the same time we are removing our trust from approximately 50% of an image which - by definition - can't be trusted. In our perspective, this would be an improvement in our work since our baseline, CIIDefence, does not trust only $735/(224 * 224) = 1.46\%$ of the total area.

For what concerns the salient areas, we are denoising it in the same way in which before we were denoising the background. An example of background inpainting and foreground denoising is shown in Figure 4.6.

Figure 4.6: Comparison between the adversarial input and the image with the inpainted background after fusing the five different inpainted parts of the *divide-et-impera* scheme (partitioned inpainting).

Also switching the inpainting area with the denoised one, the defense classification accuracy did not improve a lot compared to the baseline, even if it is now faster and less gullible.

To try to solve the problem of the heterogeneity of the mask size (which directly depends upon the object size in the image), we modified the Salient Object Detection algorithm and returned always images with the same number of $p = 3000$ most salient pixels over the other salient pixels, followed by some ablation studies around this number. This modification of the Salient Object Detection method is abbreviated as *FixedPx* variant. Lowering of the number of pixel to inpaint allowed us to use the unmodified version of the DeepFill GAN which was used also by our baseline and in addition, fixing a static number of pixels for the masked area, caused two behaviours:

1. Objects smaller than 3000 pixels returned mask which are fatter with respect to the object. This was not a problem as pictures with salient objects smaller than 3000px are rare and still objects are recovered with no problem (as the size of the image to be inpainted is small).

2. Objects bigger than 3000 pixels returned mask which does not contain them all, but only the most salient pixels of the salient area. A comparison of such behaviour is shown in Figure 4.7. In this case the shape of the salient object can not be maintained but as all the returned salient pixels lie inside the salient object, inpainting those pixels could

break some adversarial-crafted features which may be inside the most relevant area of the picture, that is the salient object.

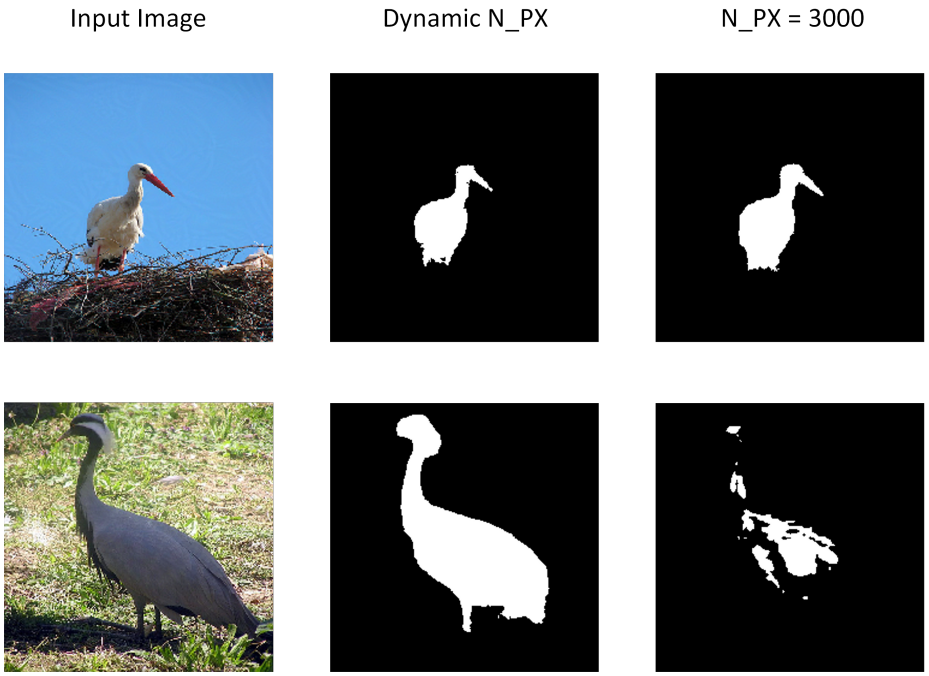


Figure 4.7: Comparison between the salient object mask generated with a static number of pixels and dynamic one. The first column from left is the original input to the salient detector, the second column is the object matched with no constraint about the number of pixels, the third column is the object matched fixing 3000 pixels.

However, even if those mechanisms represent an improvement in terms of time with respect to CIIDefence, they do not raise the accuracy significantly over the performance of our baseline. Even worst, as already said for CIIDefence in Section 3.2, according to [5] also a defense mechanism like ours, which is based only on salience detection and a deterministic inpainting and denoising, can not claim white box performance without giving full knowledge of the defense to the attacker.
In our case, an attacker with full knowledge of the defense would add its perturbations in the area not covered by the inpainting, because as already said the wavelet denoising filter can still be approximated by attacks designed to workaround non-differentiable layers, such as BPDA [19]. Thus our defense - as is - would probably fail in a full white-box scenario like the one described by Athalye, Carlini, and Wagner. Considered this, we will first investigate its performance in the classic way, to assess a fair comparison with our baseline model.

4.2 SETUP AND PREREQUISITES

For what concerns the parts of the pipeline reused from CIIDefence, no special setup nor additional prerequisite is needed except for the ones already detailed in Section 3.2.

Instead, for reproducing the correct functioning of the salient object detector [32], we used the code from Zheng in combination with its pre-trained model which are available in his GitHub repository [64].

4.3 EXPERIMENTS PERFORMED

In this section, we will first describe and then discuss the experiments related to the single saliency localization method along with their results and then we will introduce the partitioned inpainting strategy, applying it to our pipeline. We will proceed this way to introduce one novelty at a time in our pipeline to have a clear idea of what benefits and what potential drawbacks we are bringing in.

It is important to note that, even if we did not identify too much discrepancy in the comparison done in Section 3.4 between DeepFill v1 and DeepFill v2, most of the experiments mentioned here were still run twice: the first time with DeepFill v1 and the second time with DeepFill v2.

Also for these experiments, we run our tests on five different servers, each one of them equipped with a Nvidia K80 GPU and 12Gb of Graphical Memory, and an auxiliary server equipped with eight different Nvidia 1080Ti GPU each having 12 Gb of Graphical Memory too. The results that we obtained are shown next to each experiment.

4.3.1 *Saliency-based localization*

After the consolidation of the idea of replacing the CIIDefence CAMs localization with the saliency-based one while keeping unaltered the rest of the pipeline, the first step we needed to take was to find a way to assess the localization performance of the single saliency-based method with respect to the single CAMs method for localization.

4.3.1.1 *Object over a white background*

To assess the Salient Object Method functioning we tried to put the detected salient object over a completely white background to check its prediction accuracy. This experiment was only needed to test the quality of the localization done by the Salient Object Localization technique, and not the final classification accuracy, which we expect to be low compared to a full defense. Indeed, taking the portion of the image returned by the Salient Object Detec-

Figure 4.8: Example of a recognized salient object put over a white background for the assessment of the Salient Object Detection method.

tion network and trying to classify it without relying on its background, was something that would help to understand the performance of the method in presence of adversarial examples. Indeed, the Salient Object Detection already has some studies behind it [32] but those studies are executed on clean, unperturbed images.

An example of image which was submitted to the classifier in this experiment is reported in Figure 4.8. Of course, if the Salient Object Detection methods fails in identifying the foreground in many images, we would expect a step decrease in the accuracy with respect to the unmodified adversarial example.

The results are shown in the Table 4.1. Before discussing the results of the experiment we immediately notice that the Adversarial top-1 and top-5 have not changed from the values reported in Table 3.1 for the original baseline test. This is explained by the fact that once generated images with a fixed epsilon for a certain network, those adversarial examples will be reused by us for testing that network to ensure the comparability of the results across experiments.

The top-1 accuracy of this experiment is not high. Even if some adversarial perturbation which would lay in the background has been deleted, it is also

Attack	Adv Top-1	Adv Top-5	Saliency over white bg Top-1
Inception v3			
DeepFool	48.27%	88.32%	**53.58%**
FGSM	50.62%	83.73%	**53.50%**
PGD	35.78%	70.23%	**52.40%**
ResNet-101			
DeepFool	40.14%	85.18%	**39.37%**
FGSM	41.39%	83.33%	**39.20%**
PGD	30.59%	77.60%	**38.86%**
VGG16			
DeepFool	45.89%	86.88%	**32.22%**
FGSM	47.22%	84.80%	**32.17%**
PGD	42.67%	82.25%	**31.82%**

Table 4.1: Saliency assessments results over a white background on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$.

true that any kind of perturbation *inside* the salient object has been untouched. In addition, the removal of a big portion of the image (in average the background identified by this method covers over 80% of the total surface) badly influence the classification accuracy. For example, in order to distinguish a missile from a projectile, the network may need to analyze if the area near the bottom of it would present some smoke produced by the engine or not. In other words, context information is relevant to the classification task.

By testing our architecture on images which present adversarial noise and at the same time miss important features, we would expect a step decrease in the accuracy with respect to the unmodified adversarial example if the Salient Object Detection mechanism fails in identifying the salient area, because in that case we would have no clue to guess the real class of the image, having deleted it, and in addition we are left with adversarial noise. Otherwise, if the salient object is correctly identified, we expect different results based on how much adversarial perturbation is kept in the image, but in average even a small increase or a stable result could imply a good functioning of the salient detection mechanism.

Confirming again the differences in the three networks already discussed in Section 3.4, we note that the weaker of the three, VGG16, shows a drop in accuracy of approximately -15% with respect to the unmodified adversarial examples. On the other hand, more advanced networks like Inception v3 and

ResNet-101 do not exhibit such behaviour, quite the opposite, they show a stable results with increasing trends of approximately +10%. The increase in the classification outcome achieved considered the removal of features and the non-removal of the adversarial perturbation inside salient objects confirm that the Salient Object Detection network could be used in our pipeline and so it will be analyzed in depth in the proceedings of our analysis.

4.3.1.2 *Saliency vs CIIDefence*

As we assessed the performance of the Salient Object Detection mechanism executed alone and confirmed its validity for our studies, we compared its localization performance in presence of adversarial attacks with respect to our baseline, which uses the Class Activation Maps. However, it should be clear that these two methods present totally different advantages and disadvantages and thus the comparison should be performed in different settings. We will start by measuring these performance placing our analysis in the most favorite scenario for the CAMs based method.

To maintain a good degree of correspondence between the two pipelines, we used the FixedPx variant of Salient Object Detection selecting only $5 \cdot 3 \cdot (7 \times 7) = 735$ px, that is the same number of pixels identified by the CAMs. In addition, we used the same DeepFill inpainting strategy as our baseline (one-shot inpainting), thus creating an architecture where the only thing that has changed with respect to the original CIIDefence pipeline is the use of Salient Object Detection as localization strategy instead of CAMs.

As discussed in Section 3.1, the Class Activation Maps could perform a sort of coarse localization and thus we usually take only the peak maximal values of the CAMs to identify salient areas. In the example of CIIDefence, the ablation studies performed by Gupta and Rahtu suggested to take 15 squares of size 7×7 for a total of 735px. To compare the two methodologies from a fair point of view for CIIDefence we capped our saliency method in returning only the 735px most salient pixels even if the method would have the capabilities to return the entire salient object. Anyhow, as shown in Figure 4.7, the localization performed by Salient Object Detection method with fixed pixel number is sub-optimal with respect to the full method, especially for a low pixel values which makes more probable that the salient object would not be included entirely.

The results are shown in Table 4.2. As it can be seen, the Saliency mechanism suffers a lot for its 735px capping strategy and even if its performance are good and encouraging, it can't reach our baseline model inpainting such a small number of pixels. However, at this stage we can't exclude the FixedPx version of saliency, we can only state that 735px are in average a too few pixels for this strategy to succeed.

Attack	Adv Top-1	Baseline	Saliency 735px Top-1 DF1	Saliency 735px Top-1 DF2
Inception v3				
DeepFool	48.47%	**75.23**%	65.57%	67.33%
FGSM	50.62%	**76.49**%	66.09%	67.79%
PGD	35.78%	**70.02**%	61.55%	62.96%
ResNet-101				
DeepFool	40.14%	**64.06**%	54.59%	56.63%
FGSM	41.39%	**64.49**%	54.84%	57.10%
PGD	30.59%	**58.41**%	50.79%	52.66%
VGG16				
DeepFool	45.89%	**55.58**%	46.21%	48.14%
FGSM	47.22%	**55.72**%	46.44%	48.44%
PGD	42.67%	**52.74**%	43.92%	46.05%

Table 4.2: Comparison of the results obtained by using the salient object detection with 735 pixels compared with the baseline results as proposed in [1] tested on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$.

A quick check running the Saliency method with no cap on the pixel number and measuring the returned area size reported that in average the salient object size is 2500px big, with a variance $\sigma = 1000$px. This could explain the poor performance of the mechanism with 735px and lead us to an ablation study around the number of pixels for the FixedPx variant of the Salient Object Detection method to assess the full capabilities of the new localization strategy.

4.3.1.3 *Ablation studies*

Once we assessed the capabilities of our first proposed modification to the defense architecture, we put in place an ablation study in order to identify the best number of pixel to inpaint in the FixedPx variant of the Salient Object Detection method. Indeed, as shown in Figure 4.7, the number of pixel is an important hyperparameter of our method which identify only the top N most salient pixels instead of reporting the whole object shape.

As anticipated, saliency method with no cap on the pixel number reports in average a salient object size of 2200px with a variance $\sigma = 1000$px. This thing gives us 4 different choices for the parameter that is the focus of our ablation studies: we compared the performance of the full pipeline using a

localization strategy with a number of pixel from 1000 to 4000 with steps of 1000px. In this way we try pixels values in a range of $\pm$ 2σ from the mean value of 2500px, thus including around 95% of the salient object sizes of our images.

We did not use yet the partitioned inpainting mechanism, as in those range of number of pixels it would be still possible for the standard inpainting GAN to reconstruct the missing areas in just one pass. The partitioned inpainting will come into play when the area to be inpainted is so big that it would be impossible to hallucinate such a big size in just one pass.

From the results shown in Table 4.3 it is possible to see that the pixel number increase had a positive impact on the classification outcome. In particular, using a pixel size greater or equal than 2000px the top-1 accuracy has overtook our baseline results (shown in Table 3.1) of approximately +2%. The key result here does not lay in the fact that we created a method which can score a few percent points over the baseline, but instead the key takeaway is that we built a pipeline which has approximately the same results of CIIDefence, but runs 3x faster. Indeed, having replaced the CAMs with the Salient Object Detection strategy we were able to perform the localization step in 80ms instead of 2s. The remaining stages of the architecture are the same which took 1s and thus we decreased the time needed from 3s for each image to 1s per image. In doing this, we were also able to achieve a little improvement also in the classification accuracy.

From the same table it is possible to confirm once again that the classification performance does not depend too much on the specific inpainting network such as DeepFill v1 and DeepFill v2, but it is approximately equal using any GAN from Yu et al. [67, 66]. However, the trends shown in the experiment prove that the classification performance suffer more with fewer pixels than with bigger numbers and that is explained by the fact that inpainting fewer pixels we are leaving untouched large parts of the image where adversarial pixels might hide.

Those ablation studies confirms that the optimal range for the number of pixel is between 2000 and 4000. In average, we have increased the no-trust zone with respect to CIIDefence, bringing it to $3000/(224 * 224) = 5.97\%$ of the size of the total image, with respect to the $735/(224 * 224) = 1.46\%$ of CIIDefence.

4.3.2 *Partitioned Inpainting strategy*

Following the same scheme used in the previous section we will gradually introduce the partitioned inpainting strategy by assessing the performance of a pipeline which is just the same as our baseline CIIDefence except for the single building block which we are analyzing.

Attack	Adv Top-1	Adv Top-5	1000px	2000px	3000px	4000px
Standard DeepFill v1						
Inception v3						
DeepFool	48.27%	88.32%	38.48%	**75.03%**	74.84%	73.94%
FGSM	50.62%	83.73%	38.65%	**76.28%**	75.62%	73.81%
PGD	35.78%	70.23%	36.59%	70.30%	**70.32%**	68.97%
ResNet-101						
DeepFool	40.14%	85.18%	31.35%	**63.12%**	62.55%	61.32%
FGSM	41.39%	83.33%	31.43%	**63.55%**	62.91%	61.48%
PGD	30.59%	77.60%	29.44%	58.14%	**58.41%**	57.15%
VGG16						
DeepFool	45.89%	86.88%	25.01%	**54.77%**	53.51%	51.54%
FGSM	47.22%	84.80%	24.92%	**55.35%**	53.87%	51.61%
PGD	42.67%	82.25%	23.84%	**52.11%**	50.78%	48.94%
Standard DeepFill v2						
Inception v3						
DeepFool	48.27%	88.32%	43.59%	**75.42%**	75.30%	75.01%
FGSM	50.62%	83.73%	43.63%	**76.33%**	76.30%	74.91%
PGD	35.78%	70.23%	41.69%	70.09%	**70.12%**	69.95%
ResNet-101						
DeepFool	40.14%	85.18%	36.62%	**63.53%**	63.47%	62.90%
FGSM	41.39%	83.33%	36.42%	**64.22%**	64.12%	63.62%
PGD	30.59%	77.60%	33.86%	**58.69%**	59.08%	59.01%
VGG16						
DeepFool	45.89%	86.88%	28.95%	**54.91%**	54.63%	54.05%
FGSM	47.22%	84.80%	28.88%	**55.67%**	55.03%	54.16%
PGD	42.67%	82.25%	27.76%	**52.46%**	52.39%	51.60%

Table 4.3: Saliency ablation study results for different number of pixel to take as most salient ones on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$.

4.3.2.1 *CAMs and Partitioned Inpainting strategy*

As detailed in the partitioned inpainting strategy description in Section 4.1.2, this methodology has born to overcome the problem of inpainting large areas

without loss of lots of details and features. However, the localization strategy in CIIDefence is designed to take into account this problem by returning a low number of pixel, precisely 735px.

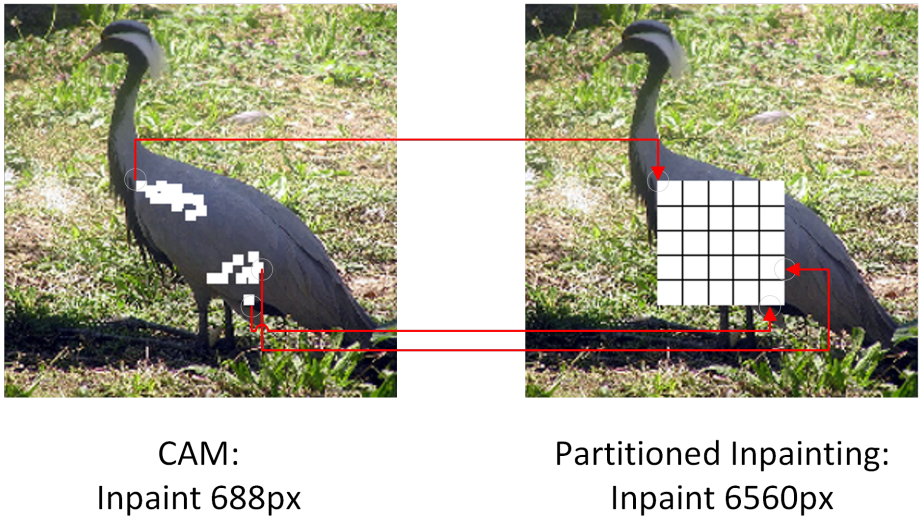


Figure 4.9: Comparison between inpainting done in CIIDefence and the inpainting done by the partitioned inpainting method guided by the CAMs. The localization idea is the same for both of them but while CIIDefence only inpaint 15 squares of size 7×7, in the proposed weakly supervised localisation strategy we inpaint the smallest rectangle containing all the pixels identified by the CAMs. In this way the number of pixel contained in the rectangle varies between images.

In order to test the potential of the partitioned inpainting strategy leaving in place the CAMs localization to account for our one-change-at-a-time approach, we inpainted not only the 7×7 tiles returned by the CAMs but instead we reconstruct-by-partitioning the whole area covered by the smallest rectangle containing all the salient pixels generated via the CAMs, as shown in Figure 4.9. The partitioned inpainting strategy divides the area to be inpainted in 25 squares (five rows of five columns) and inpaint each one singularly, aggregating them all only at the end.

The results showed in Table 4.4 for this test are encouraging as the performance using CAMs for localization and the partitioned inpainting method to reconstruct those areas have reached our target baseline model which was using CAMs for localization and standard single-pass DeepFill for inpainting.

Having reached the baseline performance changing the inpainting strategy means that, for the specific localization performed by means of CAMs, our inpainting method has the same expressiveness of the standard DeepFill and so we can continue our road to the input zero-trust using a this new strategy which for sure will help, having increased the reachable area size for

Attack	CAM with P.I. Top-1 DF1	CAM with P.I. Top-1 DF2	Baseline
Inception v3			
DeepFool	75.40%	**75.52**%	75.23%
FGSM	76.78%	**76.94**%	76.49%
PGD	**70.40**%	69.81%	70.02%
ResNet-101			
DeepFool	63.86%	**64.06**%	64.06%
FGSM	63.99%	**64.64**%	64.49%
PGD	58.13%	**58.65**%	58.41%
VGG16			
DeepFool	55.26%	**55.62**%	55.58%
FGSM	55.81%	**56.06**%	55.72%
PGD	52.77%	**52.89**%	52.74%

Table 4.4: Saliency assessments results on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$.

inpainting. As Figure 4.9 suggest we obtained the same accuracy removing trust from more than 20% of the image in average.

4.3.2.2 *Fixed px vs Dynamic px*

Having verified that our partitioned inpainting strategy has the same expressiveness as the inpainting method used in our baseline we can proceed taking a further change in our architecture. Indeed, we can exploit the full potential of the partitioned inpainting strategy to inpaint bigger areas such as the full salient object size as returned from the original Salient Object Detection as it was proposed in [32].

In addition, having identified the best number of pixel to inpaint in the FixedPx variant of the Salient Object Detection method, we were ready to compare if the usage of a dynamic area size would have brought some improvements or not with respect to the FixedPx best model. Using a dynamic pixel size area results in the full object being inpainted, but on the other hand if the salient object is too big with respect to the total input image size even for the partitioned inpainting scheme, having a fixed, carefully-chosen number of pixel might help the Generative Adeversarial Network to recover adversarially perturbed pixels in a better way.

Until now, we discovered that the Salient Object Detection mechanism was able to reach our baseline results by fixing an appropriate number of pixels to

inpaint, and that the classification accuracy decreases more rapidly inpainting lower values while decreases more slowly by fixing higher numbers of pixels. We have not run yet a comparison between the standard Salient Object Detection pipeline as presented in [32] which returns a dynamic area size depending on the salient object size and the FixedPx variant which is capped at the N topmost salient pixels.

It is true that the defended top-1 accuracy presented in the ablation studies already performs well by fixing a static number of pixel to inpaint, however there could be performance gains in sticking to the object size and no more to a predefined hyperparameter.

As we are now testing an architecture which uses a dynamic size for salient object size, we should employ the improved inpainting strategy, namely the partitioned inpainting. Indeed, it would not be possible for any GAN to hallucinate totally missing salient object just by looking at the context without any clue at all. To ensure consistency, we will use partitioned inpainting also for the FixedPx variant of the method, reason why we will not have the very same results as the previous section. Using such inpainting strategy made us compare the Dynamic version with the 3000px version, as it is true that using 3000px is slightly suboptimal with respect to 2000px but at this time we are using partitioned inpainting which does handle bigger mask sizes with no problem and thus the 3000px variant is more appropriate.

The experimental top-1 classification accuracy of such comparison is shown in Table 4.5. The obvious starting point of the discussion about these results resides in the difference between the pipeline which uses DeepFill v1 with dynamic object size and the architecture which uses DeepFill v2 with the same dynamic object size. Indeed, we can not find such difference in the comparison between the other two experiments where we compare our two GANs running the fixed-size pipeline.

The reason why, using a dynamic number of pixels, DeepFill v2 performs better than DeepFill v1 lays in the fact that in this setting the Salient Object Detection network almost always includes in the mask the whole object and from what discussed in Section 3.1.2.1, DeepFill v2 is more able to handle such situations with respect to DeepFill v1. Instead, when parts of the salient object are visible such when the number of pixel of the mask is fixed or we employ the CAMs for localization, DeepFill v1 is good in reconstructing the salient area as much as DeepFill v2, as it is possible to see in the column Fix DF1 and Fix DF2.

Following this research direction we are stuck on our initial goal of removing the trust from the adversarial images. Indeed, we proposed the Salient Object Detection to be used as localization method, decreasing the time needed while maintaining the accuracy (even improving it a bit), but as the dynamic version of such salient object detection network does not perform better than our modified one which uses a fixed number of pixel, we are not

Attack	Adv Top-1	Adv Top-5	Fix DF1	Fix DF2	Dyn DF1	Dyn DF2
Inception v3						
DeepFool	48.27%	88.32%	76.00%	**76.72%**	63.94%	67.76%
FGSM	50.62%	83.73%	76.93%	**77.22%**	64.49%	68.05%
PGD	35.78%	70.23%	71.59%	**72.16%**	61.77%	65.25%
ResNet-101						
DeepFool	40.14%	85.18%	64.50%	**65.13%**	58.02%	61.30%
FGSM	41.39%	83.33%	65.03%	**65.60%**	57.88%	61.15%
PGD	30.59%	77.60%	59.65%	**60.07%**	55.17%	58.54%
VGG16						
DeepFool	45.89%	86.88%	54.50%	**55.46%**	45.95%	47.85%
FGSM	47.22%	84.80%	55.09%	**56.04%**	46.11%	48.04%
PGD	42.67%	82.25%	51.94%	**53.05%**	44.14%	46.23%

Table 4.5: Comparison between saliency results using a fixed pixel size over a dynamic one on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$. All the pipeline used partitioned inpainting instead of the standard one with DeepFill v1.

able to remove any more trust from the images. From inpainting $1,46\%$ of the image we now are not trusting (and thus inpainting) $3000/(224 \times 224) = 6\%$ of the total image but still we leave room for attackers to exploit the other 94% of the image which is only protected by a denoiser filter which, at least in the CIIDefence paper, the attacker is not aware of.

4.3.2.3 *Background vs foreground inpainting*

We consider the FixedPx strategy an evolution of the CIIDefence architecture in terms of time and accuracy, but still we wish to follow a research direction oriented trough a low-trust models and thus we want to investigate if the lack of performance that the Salient Object Method has when using the full dynamic pixel number mask can be solved by a different inpainting strategy.

Until now, in all our experiments we have always followed the direction impressed by CIIDefence which made us to inpaint the salient object (aka foreground) and to denoise the background, except when stated differently like in the salient object over white background experiment. However, we can also decide to inpaint the background, which should contain a less amount of useful features, and denoise the foreground, which should preserve as much

as possible original clean features removing adversarial ones. In general, we have two possibilities to proceed:

- Inpaint the foreground and denoise the background: this strategy relies on the fact that the foreground (or salient object) is the most critical area of our image for its classification. To choose the correct class of the image indeed, the architecture should look at the salient object which, if adversarially compromised, may not show the features needed to the network to identify it. By inpainting and partitioning it, we try to recover such features that should increase the prediction confidence of the true class.
  Since we inpaint the foreground, we need to choose what to do with the background. The choice here is to denoise it, based on the observation that some attacks just don't care about decreasing the confidence of the true class but they add artificially crafted features elsewhere in the image to divert the network to focus in another area. Thus, the background should be cleaned from these adversarial features but as we already inpainted the salient object, we can only denoise the background. If this strategy will have success with the attacks we are testing, we should take care to test such defense also with attacks which approximate the non differentiable layers (such as the wavelet denoiser) to give full knowledge of the defense to the attacker before claiming white-box performance [19].
- Inpaint the background and denoise the foreground: this strategy is the exact opposite of the first but is born from the same motivations. Considered that inpainting the whole salient object (or most parts of it) is achieved by the state-of-the-art GANs without too much details and features (needed for the classification) we might change the strategy to the opposite. We then inpaint the whole background with a partitioned-based inpainting strategy (because of its size) and we don't care if the GAN is not able to recover in a perfect way all the details of the background, we just need that it is able to remove the adversarial perturbation from it. Of course, doing this, we should then defend the salient object from the attack and so we have chosen to denoise it, as the denoising filter maintains better the features already present (as it does not need to hallucinate new content based on the surroundings), it just need to slightly change some pixel values to be more coherent and have less adversarial variance from one another. Also in this case the considerations from [19] are valid and we need to pay attention in this sense if we will proceed in this way.

In addition, in the next Chapter 5 we will present a new inpainting method which is the core result of this Thesis and we will test the saliency based localization inpainted with this new technique.

Attack	Adv Top-1	Adv Top-5	Inp. Background DynPx	Inp. Foreground DynPx
Inception v3				
DeepFool	48.27%	88.32%	**67.88**%	67.76%
FGSM	50.62%	83.73%	**68.32**%	68.05%
PGD	35.78%	70.23%	**65.59**%	65.25%
ResNet-101				
DeepFool	40.14%	85.18%	61.20%	**61.30**%
FGSM	41.39%	83.33%	61.13%	**61.15**%
PGD	30.59%	77.60%	58.52%	**58.54**%
VGG16				
DeepFool	45.89%	86.88%	47.82%	**47.85**%
FGSM	47.22%	84.80%	47.99%	**48.04**%
PGD	42.67%	82.25%	**46.42**%	46.23%

Table 4.6: Comparison between saliency results using a denoising filter on foreground and inpainting the background versus the opposite strategy on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$. All the pipeline used partitioned inpainting instead of the standard DeepFill.

We have reported in Table 4.6 results of this experiment. We can see that for all the networks and all the attacks, we get very similar results even by totally changing the inpainting strategy. An image with an inpainted background and a denoised salient object would appear to be different both in terms of human perception and in terms of clean features with respect to an image which has a denoised background and an inpainted foreground. Yet, we see that in average the classification network has near the same performance on these two different kind of images.

However there is a substantial difference in the two kind of images: while in the input with the foreground inpainted we erase and restore an area with a size of about 2000px−3000px that is around 6% of the total image, in the picture with the background inpainted we cancel and recover the opposite area, that is $100\% - 6\% = 94\%$, maintaining the same performance.

We discovered a trade-off between the size of the area to trust and the importance of the information inside that area. It is possible to inpaint the 94% of the image (thus trusting only 6% of it) if in that area we do not have too many features of the object to be classified (background). At the opposite, if we want to inpaint a critical area for the classification outcome that is the salient object, we should put our trust in huge parts of the image, such as the whole background. Of course one method could perform better than other

when subject to specific attacks. For example, in the paper "Mask-guided noise restriction adversarial attacks for image classification" [50], Duan et al. propose an attack which concentrate all the adversarial perturbation inside the salient object. Needless to say, for this specific attacks, the method which should outperform the other is to inpaint the salient object and denoise the background. But as in a white box scenario we are defending from unknown types of attacks, we can not favorite a specific inpainting method on this basis. All the attacks that we have chosen to test our architecture, which are general well-know attacks in the literature, place their perturbation all over the image without preferences explaining in this way the quite similar performance of the two different inpainting methods.

Such trade-off between the size of the area to trust and the importance of the information inside that area will be solved in the next chapter.

5

DEFENSE BY MASSIVE INPAINTING

Since the beginning of our studies we followed a research direction oriented in removing as much trust as possible from adversarial images because, as the name suggest, those are images which were artificially manipulated by an adversary at the only purpose of fooling a classification network to make a wrong prediction.

However, the concept of trust related to an image is broad, and we want to give a categorization to it by dividing such concept in two distinct categories:

- Trust pixel-level values: this category refers to whether we should take for granted the single individually-considered RGB values inside a pixel. Those values ranges from 0 to 255 and identify the intensity of a channel in the pixel. An adversary has the power to alter those values in such an imperceptible way that new adversarial features appear at a macroscopic level and real class features are broken at microscopic level. Therefore image has some pixels which values are changed of a few points, thus those variations are not easily noticeable by a human, but those pixel-level changes can be singularly used to break real features of the true class in such a way that the network lower his confidence in the real class. In addition, if those microscopic variations are correlated one another as it is in adversarial attacks, the resulting adversarial features from those variations are superimposed over the real image with a low intensity. Even if those pixel-level variations are not clearly distinguishable by humans, the network which should classify the image often get confused by the presence of such human-invisible perturbations that exhibits adversarial features and thus the input is often misclassified.

- Trust macroscopic level features: this level of trust in the image is related to the trust that we should have in the features present in the image. Non-adversarially trained neural network can not discriminate the bounty of a feature by only looking at its relative intensity over the background because those networks were trained to make classification guess, not other tasks such as adversarial recognition. Classification networks acquire all the features shown in the input image and then assign them an importance score accordingly to different factors, such as the mutual presence of some features which may resemble a class that the network knows from train time. The problem with macroscopic level perturbations is that we can not delete it as we would with pixel-

> level RGB values: for example, microscopic changes can be achieved by looking at neighborhood pixels and increasing their relative co-existence probability, as Bayesian denoisers do. However, the concept of adversarial feature or macroscopic perturbation for a neural network still arise from the microscopic presence of certain patterns in pixel-level values thus to eliminate the trust in macroscopic level features we would still need to alter the microscopic level RGB values, but in such a way that the high-level coherence between them is preserved.

We discovered that the method proposed in Chapter 4 presents a trade-off between the size of the area to trust and the importance of the information inside that area. Rephrased, the trade-off is between the pixel-level trust and the feature-level trust. Indeed, we can mess up with many pixel as long as they are not part of important features, while we can only modify a small number of single pixel values if we are operating in a zone rich of features which are important for the classification outcome, such as the foreground or salient object area.

This trade-off however, is something connected to the method. Architectures such as Pixel-Deflection [3], CIIDefence [1] and our improved saliency-based localization and inpainting [Chap. 4], all suffer of this trade-off.

In order to avoid the presence of this trade-off between inpainting a small number of pixel of the foreground, and inpainting a big number of pixels of the background, we would need a way which allows us to do both: this technique is called Massive Inpainting.

5.1 DESCRIPTION AND METHODOLOGY

We propose a novel defense strategy which we call Defence by Massive Inpainting in which we alter the pixel-level values present in the input image in such a way that macroscopic-level adversarial features are removed and non-adversary macroscopic-level features are maintained. Indeed, as we wish to migrate to a zero-trust model, we propose a way to defend from adversarial attack which modify *all* the pixels of the input image, regardless of it being a salient pixel or a background pixel.

To the best of our knowledge, the only research work which applies inpainting to adversarial attacks is [28], the Erase-and-Restore attack detection mechanism. In this work, Zuo and Zeng randomly draw a percentage of pixels of the input image, then they inpaint only those pixels and the resulting image is compared back with the original image, in order to detect if the inpainting has changed or not the input image. If the image has been modified a lot, the input picture is considered to be adversarially perturbed. However, in this Thesis we are proposing a defense mechanism and not a

detection mechanism and in addition, while they inpaint a subset of pixels, we always inpaint the 100% of input pixels.

A natural consideration to make is that, as we are not anymore interested in knowing where the salient objects and the most influent areas for the classification are, we can now drop the localization stage of our pipeline. Completely dropping this stage will increase the architectural speed even more than what we achieved by introducing the saliency-based localization and in addition it removes an exposed attack surface that in a full white-box scenario the adversary can exploit. By giving full knowledge of the defense to the adversary [5], it is no more possible from him to integrate the localization defense mechanism in its adversarial example generation pipeline to avoid well-defended areas and to find the best location for his perturbation, because the localization stage was removed and now it does not exist anymore. By inpainting all the pixels of the image, the only remaining possibility that an adversary has thanks to the full knowledge of the defense architecture would be to target the Generative Adversarial Network which performs the defensive inpainting. Attacks on the GAN are not common in the literature as to the best of our knowledge, all the works mentioning GANs and Adversarial Attacks are *using* an adversarially-trained GAN to generate Adversarial Examples, but are not generating Adversarial Examples which target a specific GAN to lower its reconstruction accuracy. The process of creating a new adversarial attack specific for GANs is more difficult than using the common attacks already known in the literature and just avoid the inpainted zones in order to place the adversarial features in the denoised area, as it would be possible for CIIDefence. However, we will remove also the possibility to exploit the GAN in an adversarial attack (which is the only remaining added attack surface) by totally randomizing the way in which the GAN operates.

The turning point for the massive pixel reconstruction came from the reading "Reconstruction by inpainting for visual anomaly detection" [43]. In this work, Zavrtanik, Kristan, and Skočaj propose a novel architecture applied to anomaly detection of objects and visual quality control which uses inpainting as one of their pipeline stages. Using their inpainting method as a defense against adversarial attacks has, to the best of our knowledge, never been studied in literature.

5.1.1 *RIAD*

Reconstruction by Inpainting for visual Anomaly Detection (RIAD) [43] addresses the task of anomaly detection of defects in objects by only analyzing their pictures and reduces the problem to a classification and localization of areas of an image which deviate from their normal aspect, which thus may have a defect. Popular approaches for solving this problem involve the

training of an auto-encoder (which acts as a denoising filter) and computing the similarity between the original image and the denoised output. If the similarity is high, it means that the denoiser has not performed many modifications and thus the object in the picture is considered to be defect-free. On the other hand, if the similarity is low, it means that the denoiser has removed something from the input image (probably the defect) and thus the object should be considered as defective and inspected or discarded.

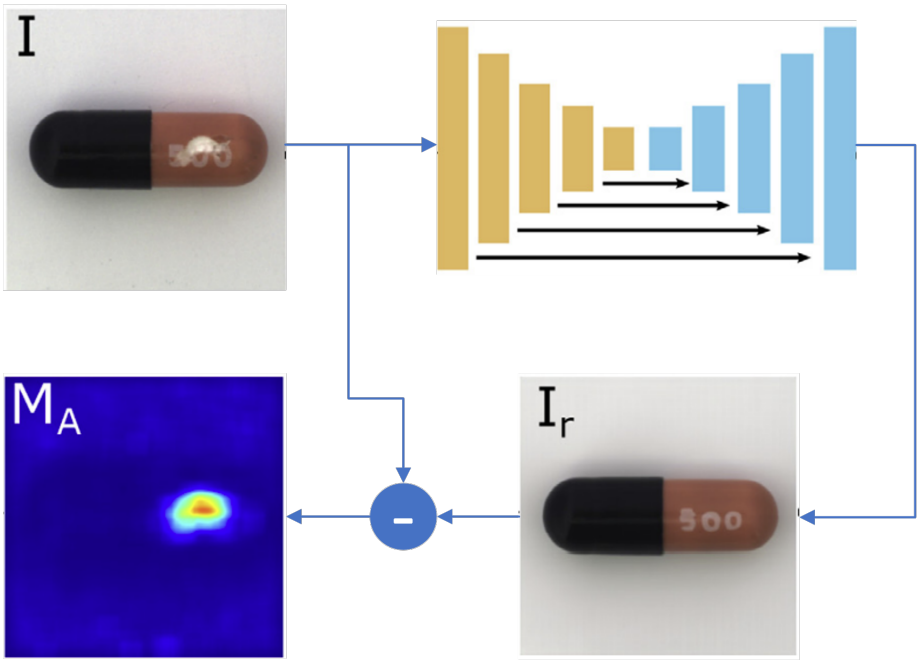


Figure 5.1: General functioning mechanism of the full RIAD pipeline. The input image is massively inpainted and then compared back to itself to identify anomalies in the object.

However encoder-decoder networks, even if trained on defects-free images, were proved to generalize well also to anomalies, thus reducing the detection capabilities [43]. Based on this assumption, Zavrtanik, Kristan, and Skočaj formulate the problem of visual anomaly detection as a reconstruction by inpainting problem. They propose a massive inpainting strategy which is able to reconstruct all the pixels of the input image and afterwards, as other methods do, they compare the inpainted result with the input picture to check if anomalies were present in the object, as shown in Figure 5.1.

5.1.1.1 *Inpainting strategy of RIAD*

For what concerns the adversarial attacks application that we will explain, the most interesting stage in their work is the mechanism which they employ

to erase and restore every single pixel of the original image maintaining an high level of detail.

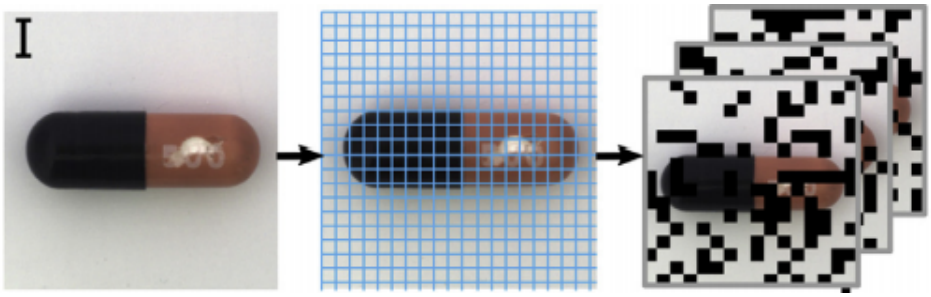


Figure 5.2: General functioning mechanism of the inpainting strategy used in the RIAD pipeline. The input image is randomly masked by small tiles in such a way to uniformly distribute all the tiles between the N masks.

The mask generation strategy used in RIAD creates a fixed number of masks and then it computes the percentage of image area to hide in each of these mask in order to cover the full image area. For example, if we choose to generate N masks it is clear that each mask should contain 1/N of the total original image area. In addition, instead of choosing big adjacent areas to hide in the same mask, they randomly draw the percentage of the original image area tile by tile, as shown in Figure 5.2.

The inpainting strategy of RIAD uses a grid which tiles can be as small as they want thanks to the fact that the tiles dimension does not influence the number of inpainting runs. In addition, as the task that they are solving is related to anomaly detection and the defects they are detecting could space from a few pixels up to 16px and more, they use different grid sizes and aggregate all the inpainting results performed with the different grids at the end, by taking the pixel-wise average.

In particular, Zavrtanik, Kristan, and Skočaj decided to generate for each input image 4 different inpainted images respectively with a tile size of 2px, 4px, 8px and 16px. After all the inpainted outputs are ready, the final image which will be compared back to the original is created by taking the average pixel by pixel of those 4 images.

An example of the mask generation which uses 3 inpainting runs for each output and different tile sizes is shown in Figure 5.3. The motivation behind the need of such strategy in RIAD is that by aggregating different scales of tiles in the masks, the resulting image is more detailed thanks to fine-grained masks, and at the same time it is more smooth thanks to the coarse masks. In addition, it is sure that both smaller defects and bigger defects were inpainted, thanks to the multi-scale tile sizes which better cover more adjacent area in the input image.

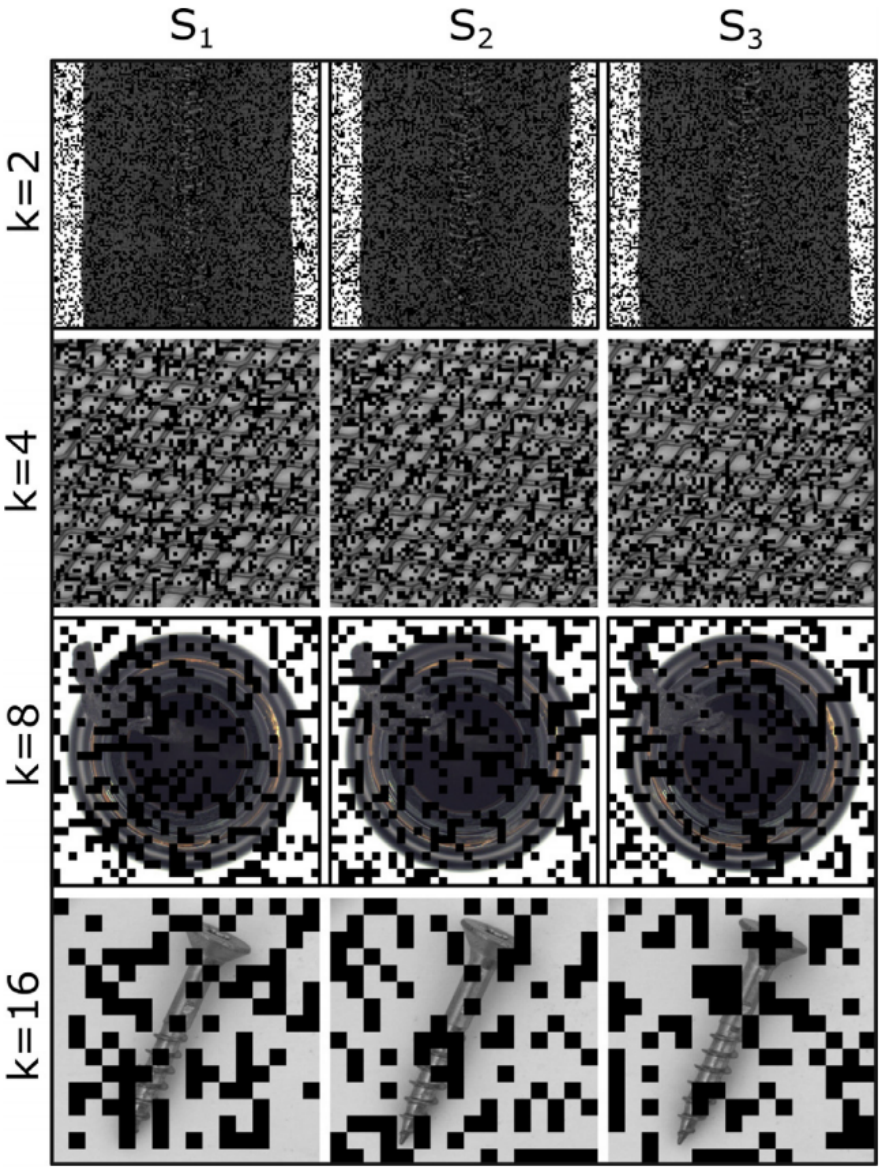


Figure 5.3: Example of the RIAD inpainting strategy taken from [43] applied to an input image masked by $N = 3$ disjoint sets of inpainting regions with four different squared mask tile of sizes 2px, 4px, 8px, 16px. In this case it needs 3 inpainting runs to reconstruct a single image with a fixed tile size, and 4 different images to aggregate at the end, for a total of $3 \cdot 4 = 12$ inpainting runs.

The inpainting mechanism which they employ is based out of the well-known U-Net architecture [56], which they trained using a multi-scale gradient magnitude similarity (MSGMS) loss [51] and a structured similarity index (SSIM) loss [53]. The choice of using those losses that penalize structural differences between the reconstructed regions and regions belonging to the

original image is related to the fact that the standard per-pixel L_2 loss which is commonly used assumes independence between neighboring pixels, which is often incorrect.

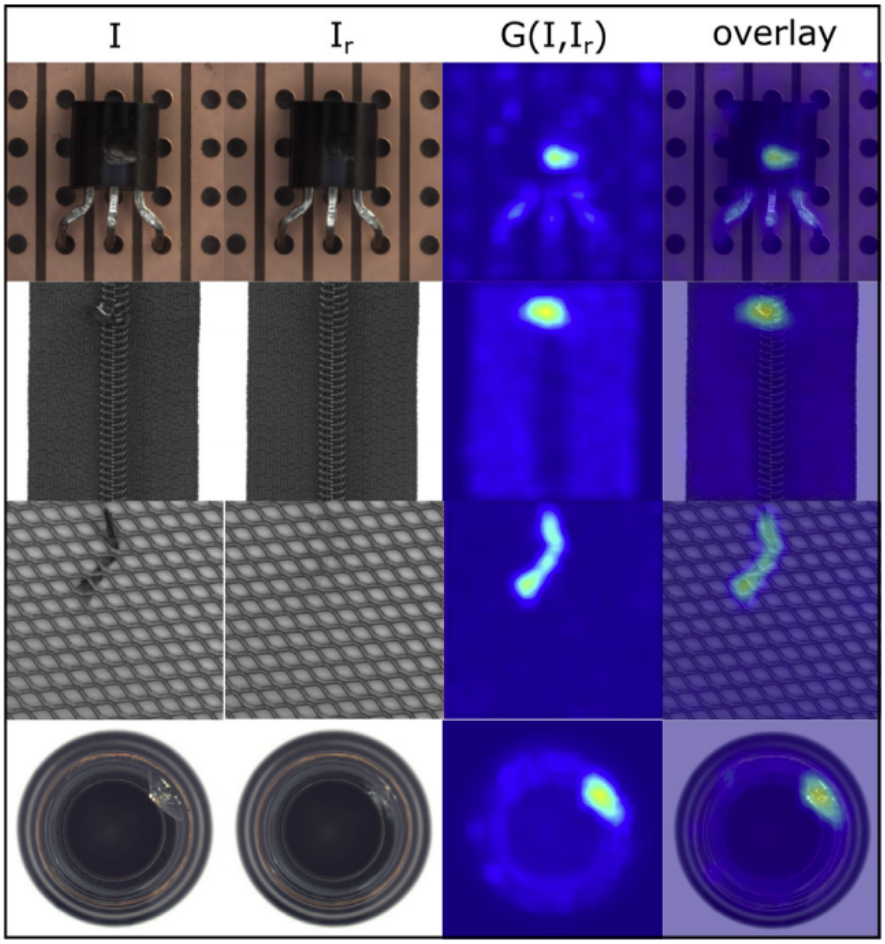


Figure 5.4: Example of the defect localization performance on different images which the original RIAD method is capable of achieving. Starting from the leftmost column we have the original image, the full inpainted image, the localization mask and the overlay of the mask over the original image.

The final result of the full RIAD method is shown in figure 5.4. In this Thesis we are interested in the specific inpainting strategy that they employed, that is the multi-scale grid sizes with random pixel level inpainting via disjoint sets, and we will experiment if and how this methodology could help in defeating adversarial attacks. We will implement this strategy using our already analyzed DeepFill GAN and thus we will not use their U-Net architecture nor any other part of their pipeline which we believe is designed for the specific task of anomaly detection.

5.1.1.2 *Benefits of the inpainting strategy of RIAD in our pipeline*

Until now our defense strategy was stuck at inpainting at most 25% of the input image with the standard DeepFill GANs which we increased to approximately 90% using the partitioned inpainting strategy, provided that the quality of the inpainting would be low and thus excluding from the inpainting process the key features for the classification.

We recall that our partitioned inpainting mechanism could improve its level of detail in the inpainting result if we set a smaller size as base area in our *divide-et-impera* approach. The method proposed in Chapter 4 divides the input mask of the inpainting GAN in 5 different rows each composed of 5 different columns, for a total of 25 tiles. Each tile is then inpainted singularly for a total of 25 different runs of the inpainting network in order to fully reconstruct the original image.

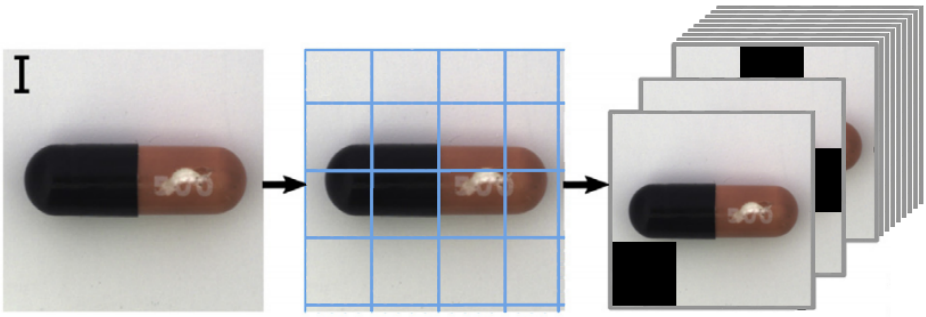


Figure 5.5: The core idea behind the mechanism of the full partitioned inpainting scheme. The input image is masked one tile at a time, thus needing N^2 different inpainting runs, where N is the number of tiles in a row or column.

Without using RIAD, if we want to increase the detail of the inpainting, we can lower the tile size of the Partitioned Inpainting method by creating a 10×10 grid (thus using 24px $\times$ 24px tiles), but then we would need to perform $10 \cdot 10 = 100$ different inpaintings just to reconstruct a single input image. An example of our partitioned inpainting technique used to massively inpaint an image is shown in Figure 5.5: it is possible to see that this method generates big tiles which badly influence the inpainting results and the reduction of their size increases quadratically the number of needed inpaintings.

Instead, integrating the inpainting strategy used by RIAD in our pipeline has at least two evident benefits:

- By using masks made of small tiles, it is possible to cover the whole image surface without influencing too much a single area of it. For example, inpainting an area of 56px $\times$ 56px $=$ 3136px creating a single adjacent black hole creates much more reconstruction error with respect

to removing 1 pixel every 4 for the whole size of the image. Indeed, in the latter case, the inpainting GAN has, for each pixel, 3 neighbors which can be used to infer the missing one. Instead, by creating the single large hole of 56px × 56px the inpainting GAN loses some of its expressiveness when it comes to hallucinate the inner pixels of such hole.

- In addition, fixing the number of inpainting runs is useful when it comes to the time required to reconstruct the entire image. Increasing the size of the Partitioned Inpainting method of 1 column and 1 row results in a squared increment of the number of the inpainting needed. Suppose for example to increase the partitioned inpainting grid from a size of 5 × 5 (25 total inpaintings) to a size of 6 × 6 (36 total inpaintings), there would be a 44% time increment even for the smallest possible variation. Instead, by using the strategy used in RIAD it is possible to fix a static number N of inpainting runs and have N different masks, each one with 1/N-th of the image which is randomly masked. In such a way we can fix any number of inpainting runs we require without caring of it being a quadratic number, a non-prime number or any other constraint.

5.1.2 *The functioning of the defense*

The defense mechanism simplifies a lot our Defense by Massive Inpainting proposition but it should be noted that the complexity resides in the way in which we defend, and not in the number of defense stages in the pipeline.

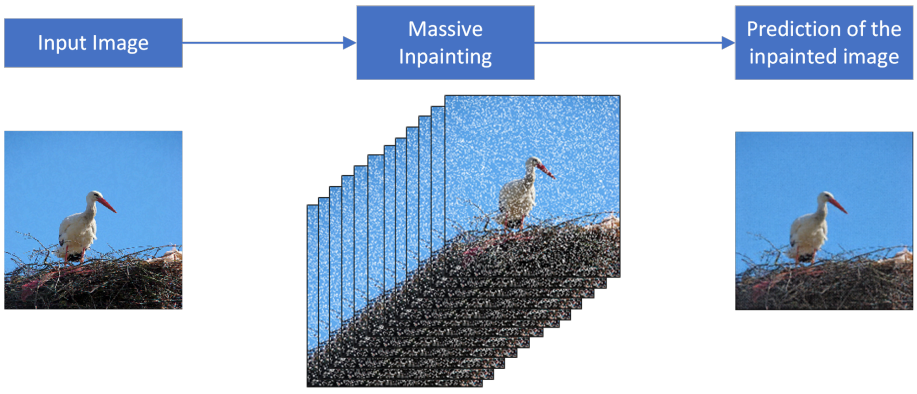


Figure 5.6: Full architectural scheme of our Defense by Massive Inpainting.

As shown in Figure 5.6, given the input image we randomly create N different pixel-level masks to be used for the massive inpainting and run N times the DeepFill architecture, each time with a different mask. Once

finished, we take from each DeepFill run the set of inpainted pixel and we aggregate them trough all the runs to reconstruct the input image with all the pixel inpainted. Finally we submit this image made of only inpainted pixels to the classification network, eventually adding a denoiser.

If we would randomly select a subset of pixels to inpaint as in [28], it would be possible to state that there can happen *lucky* inpainting runs where the architecture has randomly selected many adversarial pixels and few original ones, thus removing a lot of adversarial perturbation and facilitating the prediction. However in our mechanism, we inpaint the totality of the pixels provided as input picture, thus the final image that will be predicted by the classifier will be constituted of only inpainted pixels: no original pixel is left, we reached a zero-trust defense strategy.

Inpainting every pixel of every image result in four possible inpainting cases to be handled by the GAN:

- **Reconstructing an adversarial pixel by inferring it from its adversarial neighbors.** The single adversarial pixel we are inpainting has been erased. In order to reconstruct it, the GAN is looking to other adversarial pixels. As the inpainting network is obviously not an adversarial attack, the inpainting result would not be the perfect pixel-level adversary values as it was previously. However, the Generative Adversarial Network is not even able to guess the perfect original non-adversarial values of the pixel if it does not have any non-adversarial clue. The most probable result of this situation is noise: the reconstructed pixel will contain a value that is near the original one but also near the adversarial one, thus the pixel can be considered no more an adversarial one, but still it can challenge the classification network. In natural images, we can expect that the channel and spatial correlations are specific, and that real pictures have all similar distributions, which is the motivation behind the good generalization of generative networks to unseen images. A non-adversarially trained GAN fills the masked pixels by using its ability to model natural images correlations. However, since an adversarial example is not a natural image, the reconstructed pixel would follow a distribution which is different from the one used to attack the GAN, thus the adversary pixels values are corrupted. These situations decrease the adversariety of the image and increase the noise, thus are helpful.

- **Reconstructing a non adversarial pixel by inferring from its adversarial neighbors.** Inpainting 100% of the input pixels would also include erasing and reconstructing pixels that were left as is by the attacker, but that we still do not trust. In case a non-adversarial pixel is deleted and the neighbors which are considered by the GAN to infer the missing one are adversarial, the situation is similar to the previous described

one, and the true pixel is replaced with noise. These situations decrease the image similarity with the original input and increase the noise, thus are not helpful.

- **Reconstructing an adversarial pixel by inferring from its non-adversarial neighbors.** In this case the adversarial pixel is deleted and the GAN needs a way to infer the values for the deleted pixel. As the focus of the inpainting network is on original true pixels, it is probable that the reconstructed pixel is very similar to the original one thanks to its correlation with their original neighbors. These situations decrease the adversariety of the image and increase the similarity with the unmodified image (good).

- **Reconstructing a non adversarial pixel by inferring from its non adversarial neighbors.** in this case it is probable that the GAN would reconstruct the pixel as it is or at least very similar to it. These situations do not decrease nor increase any metric, thus are neutral.

We have always kept in mind the problem stated in [5] about the big number of defense strategies claiming white-box performance without giving full knowledge of the defense architecture to the adversary throughout all our research.

The defense strategy that we are proposing is designed is such a way that there would be no improvements for the attacker to know how it works or not. Indeed, we removed the localization stage which could be exploited by adversaries and the only remaining weak point in this sense was the inpainting stage.

To overcome possible adversaries which, in a full white-box scenario, would leverage our GAN to make its inpainting results worse than the non-inpainted adversarial image (attacks on the GAN), we used a randomized mask generation procedure. Instead of assigning each pixel to one of the N masks in a deterministic way, this choice is done in a random way following a uniform probability distribution. In such a way, two different classification runs of the same image would have slightly different inpainted results thanks to the random way in which the mask are generated. In this way, the adversary does not know which masks the GAN will use to inpaint the image and thus the adversary can not generate image which confuse the GAN maintaining a low perturbation norm. The only feasible strategy for the adversary is to increase the adversarial noise in each pixel of the image but doing this over a certain threshold would violate the basic assumption of adversarial attacks: the adversarial examples should be indistinguishable by a human from the clean image version.

Notice that the white-box threat model gives to the adversary full access to the neural network classifier architecture and weights, but not test-time

randomness (only the distribution). In this sense, the adversary can not know how the mask submitted to the GAN are made and which pixels they contain. What it can do, is to enumerate all the possibilities, that thanks to the uniform distribution, have all the same probability. The possible masks are the combination of $224 \times 224 = 50176$ pixels in class $224 \times 224/N_{IMG}$. In case $N_{IMG} = 12$, these are:

$$\binom{50176}{4181} = \frac{50176!}{4181! \cdot (50176 - 4181)!} = \frac{4.77 \cdot 10^{214063}}{9.92 \cdot 10^{13326} \cdot 5.38 \cdot 10^{194488}} \approx 10^{6250}$$

Randomness-based defenses can always be defeated by enumerating all the possibilities, but usually stochastic gradients or random weights have much lower combinations, because each one of them would require a retrain. In our case instead, brute forcing all those mask possibilities is not feasible.

Having secured all the attack surfaces that would be exposed in a full white-box scenario as described in [5], we can now focus on standard white-box attacks and still comply with Carlini et al.'s policy.

5.2 SETUP AND PREREQUISITES

The functioning of the architecture is based only on massive inpainting and reconstruction. For the inpainting stage, we took inspiration from [43] and implemented our own version of the inpainting architecture which is based on the same RIAD mask generation and uses the DeepFill GAN for the inpainting. The only required prerequisite is to use the official implementation of DeepFill from [67, 66] and its pre-trained models on Places2 [34].

As for the other experiments, we run our tests on five different servers, each one of them equipped with a Nvidia K80 GPU and 12Gb of Graphical Memory, and an auxiliary server equipped with eight different Nvidia 1080Ti GPU each having 12 Gb of Graphical Memory too. The results that we obtained are shown next to each experiment.

5.3 EXPERIMENTS PERFORMED

In this section, we will describe and then discuss the experiments related to the Massive Inpainting defense method along with their results. We will start by identifying the best hyper-parameters for our method, then we will compare our best model with our baseline and with our improved Saliency Based Localization and Inpainting, and finally we will assess the consistence of the results across different values of epsilon.

We will not cover experiments performed using the BPDA [19] attack because the proposed defense strategy do not use any kind of gradient obfuscation technique. The real gradient is freely available to all the attackers

and thus there is no need to test our method with the Backwards Pass Differentiable Approximation attack strategy.

5.3.1 *Ablation Studies*

The first thing that we want to assess in our proposed defense strategy is the hyper-parameters selection. Indeed, we will study some method variations and by running the ablation studies before the result discussion and before the comparison with other defences, we can follow a waterfall approach using in each subsequent experiment the best hyper-parameters from the previous. This strategy will allow us to reduce the number of runs per single experiment while maintaining full generality. However, for the experiments executed before having run the ablation study to identify the best tile size, the previous ones (i.e. the one to exclude the usage of denoiser) were executed with tile size $= (1 \times 1)$. Instead, before the ablation study about the number of images, the previous ones were run with $N_{img} = 12$ as suggested by the original DeepFill paper.

All the ablation studies for our hyper-parameters are executed at $\varepsilon = 1.5$ which was the upper bound limit of our baseline tests in Chapter 3 and was considered as *high* in Chapter 4 while discussing the results of Saliency Based Localization and Inpainting. The motivation behind this choice is that we need to identify the best hyper-parameters and strategies to be used at high epsilons if we want to propose a new defense strategy which should compete with other state-of-the-art approaches.

5.3.1.1 *Usage of denoisers*

The first immediate question that we posed ourselves was related to the usage of denoisers. Denoising filters have been used since the beginning of this Thesis to try to remove some adversarial perturbation in areas of the image which was not covered by the inpainting. Such usage has been proved to be weak [19] with respect to full white box attacks such as BPDA, and we already dropped this usage.

However, it is still possible to employ denoising filters for the original task they were created: removing the (non-adversarial) noise from an image. As already said in Section 5.1.2, the RIAD inpainting strategy removes some adversarial perturbation at the cost of adding noise. We can notice this behavior in Figure 5.7 and more in detail this is shown in Figure 5.8.

It is important to consider that denoiser architectures have some limitations and drawbacks. The most impacting drawback for this task would be the potential loss of details that denoising filters cause in the image in order to smooth pixels values, as shown in Figure 5.9. It is possible that the smoothness

Figure 5.7: Differences between the RIAD-like inpainted result and the original input.

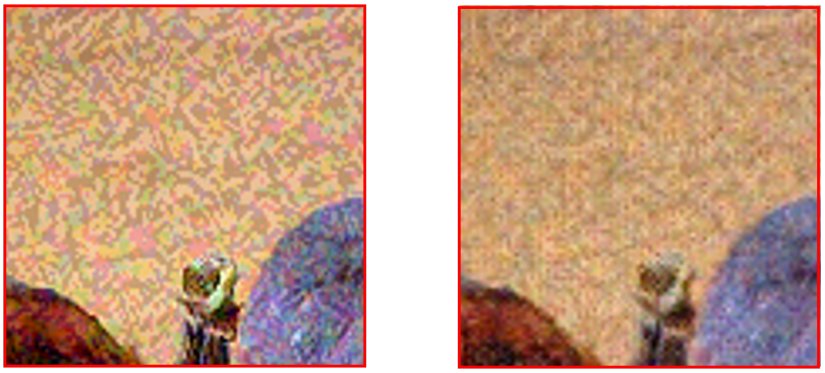


Figure 5.8: Detail of Figure 5.7 to highlight the changes performed to the adversarial perturbation by the inpainting GAN.

added by such filters is not useful for the classification architectures but it is just more eye-appealing for us.

The denoiser that we have tested is the same used by our baseline, that is the wavelet denoising filter implemented in the *skimage* library with automatic σ_{noise} estimation in YCbCr color space.

In order to verify such behaviour, we run an experiment with the goal of discovering if a denoising filter applied after the inpainting could help the classification model in better recognizing the true class. The results are shown in Table 5.1.

Inpainted Image Denoised Image

Figure 5.9: Possible improvement reached by denoising the output of the RIAD-like inpainting GAN.

Attack	Adv Top-1	Adv Top-5	Denoised DF1	Clean DF1	Denoised DF2	Clean DF2
Inception v3						
DeepFool	34.99%	78.18%	86.83%	**88.62%**	85.08%	87.67%
FGSM	35.55%	67.79%	83.08%	**84.40%**	76.87%	76.42%
PGD	3.61%	11.86%	86.23%	**88.01%**	84.21%	85.80%
ResNet-101						
DeepFool	15.15%	68.63%	77.32%	**77.91%**	69.41%	68.93%
FGSM	15.33%	55.21%	**70.23%**	70.09%	57.52%	54.58%
PGD	1.57%	30.33%	76.30%	**76.57%**	65.58%	62.64%
VGG16						
DeepFool	14.49%	69.56%	53.93%	**54.22%**	40.99%	38.04%
FGSM	12.55%	47.33%	**41.73%**	40.90%	30.11%	26.66%
PGD	3.95%	36.14%	**44.71%**	43.94%	26.02%	20.49%

Table 5.1: Ablation studies over the usage of a wavelet denoiser filter on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 1.5$.

As we have not yet assessed if it would be better to use DeepFill v1 or DeepFill v2, we used both and we compare the results here. We can see that with Inception v3, there is no improvements using a denoiser nor with DeepFill v1 nor with DeepFill v2. In addition, the performance of the denoiser filter with DeepFill v1 are better than the performance with

DeepFill v2. However, using a denoiser filter with ResNet-101 or VGG16 as classification models has similar performance with DeepFill v1 and even better performance using DeepFill v2. Along with DeepFill v2, the usage of such denoiser successfully increases the prediction accuracy, but it is worth to be noted that the performance with DeepFill v2 in such situations are low and the increase using the denoiser does not overtake the clean accuracy of DeepFill v1 without denoiser.

As a result, we think that using a denoiser does not improve our defense mechanism and then we will drop the adoption of a denoiser in such defense strategy.

5.3.1.2 *GAN performance: DeepFill v1 vs DeepFill v2*

The second experiment addresses the differences in the accuracy between DeepFill v1 and DeepFill v2. As DeepFill v1 is explicitly designed for squared masks while DeepFill v2 is explicitly created for all other mask shapes, switching from a free-form mask generation (as in Saliency Based Localization) back to masks which are composed only of squares would then suggest us to evaluate DeepFill v1 as potential candidate as the Generative Adversarial Network of our inpainting stage.

Attack	Adv Top-1	Adv Top-5	DF1	DF2
Inception v3				
DeepFool	34.99%	78.18%	**88.62%**	87.67%
FGSM	35.55%	67.79%	**84.40%**	76.42%
PGD	3.61%	11.86%	**88.01%**	85.80%
ResNet-101				
DeepFool	15.15%	68.63%	**77.91%**	68.93%
FGSM	15.33%	55.21%	**70.09%**	54.58%
PGD	1.57%	30.33%	**76.57%**	62.64%
VGG16				
DeepFool	14.49%	69.56%	**54.22%**	38.04%
FGSM	12.55%	47.33%	**40.90%**	26.66%
PGD	3.95%	36.14%	**43.94%**	20.49%

Table 5.2: Ablation tests over the two different DeepFill versions executed and tested on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 1.5$.

The results of the comparison between DeepFill v1 and DeepFill v2 are shown in table 5.2. We can see that DeepFill v1 has more stable performance

over the three different network architectures tested as its performance ranges from 41% to 89% while the second generation of DeepFill has the accuracy ranging from 20% to 88%.

In addition to the considerations done in the previous chapters about the physiological accuracy drop of VGG16 and ResNet-101, we also note that DeepFill v1 has less variance over the different attacks. Fast Gradient Sign Method (FGSM) seems to be the most challenging attack causing a drop of accuracy of 12% in average with DeepFill v2 but that drop reduces to a decrements of only 5% in average with DeepFill v1.

As the performance using DeepFill v2 are similar or worse with respect to the performance of DeepFill v1, we will continue our studies using only DeepFill v1, except when explicitly stated otherwise such in case of doubts or further experiments.

5.3.1.3 *Area of the image to inpaint in each GAN run*

One important ablation study performed is the one regarding what extension of the image we need to mask in order to obtain good performance. We recall that differently from our Partitioned Inpainting strategy, RIAD propose to fix a certain percentage p over the full image of the maximum area to inpaint and then run the GAN $1/p$ times.

Until now we have followed the advice wrote in the original DeepFill v1 paper [66] which states that their GAN is optimized to handle masks covering up to $1/12$ of the total area size. Accordingly, for each input image we run DeepFill a number of $N = 12$ times by randomly masking the image following a uniform distribution with $\bar{p} = 1/12$.

However, we challenged our proposed defense about the possibility to further improve the performance by reducing the total masked area at each DeepFill run following the idea that the GAN might infer pixels more accurately if it need to do it more times and with more available pixels to use for such inference.

The results are shown in Table 5.3. It is possible to note that there is no predominant best hyperparameter and this is due to the fact that in some networks which may have space for improving (e.g. Inception v3), the performance rises as the rise of N but not as fast as N. A possible explanation to this resides in the fact that DeepFill might already have all the needed information to infer missing pixels yet from $N = 12$ and by decreasing the total mask area (thus increasing the available information for the GAN), DeepFill may just ignore it or even worse, it may be confused as the correlation between many pixels is not the same as in the original image because these are adversarial examples, thus adversarially de-correlated. This adversarial de-correlation could be the explanation of the slightly decrease in the performance of ResNet-101 and VGG16 as N rises. Contrarily of Inception

Attack	Adv Top-1	Adv Top-5	N = 12	N = 20	N = 28
Inception v3					
DeepFool	34.99%	78.18%	88.62%	89.12%	**89.21%**
FGSM	35.55%	67.79%	**84.40%**	83.67%	83.27%
PGD	3.61%	11.86%	88.01%	88.24%	**88.30%**
ResNet-101					
DeepFool	15.15%	68.63%	77.91%	**78.52%**	75.93%
FGSM	15.33%	55.21%	**70.09%**	68.51%	67.44%
PGD	1.57%	30.33%	**76.57%**	76.20%	73.58%
VGG16					
DeepFool	14.49%	69.56%	**54.22%**	52.76%	49.98%
FGSM	12.55%	47.33%	**40.90%**	38.53%	36.03%
PGD	3.95%	36.14%	**43.94%**	40.05%	37.44%

Table 5.3: Ablation studies of different total masked area per single GAN run on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 1.5$.

v3, these networks could be more sensitive to the worse GAN outputs in presence of too much non-correlated information and may fail the prediction.

5.3.1.4 *Tile size to consider for the inpainting mask*

The final and most important ablation study regards the size of the tiles which compose our masks. Indeed, the original proposition of RIAD for anomaly detection suggest to reconstruct the image 4 different times:

1. Perform N inpainting runs using a tile size of (2×2) pixels
2. Perform N inpainting runs using a tile size of (4×4) pixels
3. Perform N inpainting runs using a tile size of (8×8) pixels
4. Perform N inpainting runs using a tile size of (16×16) pixels

After these $4 \cdot N$ inpainting runs RIAD states to aggregate the 4 different output images by taking the pixel-level average.

However, this procedure was designed for their anomaly detection task, and do not applies well to our defense strategy. Indeed, we need to take into consideration both macroscopic-level adversarial features but also pixel-level adversarial perturbations. To handle this situation and to experiment with

Figure 5.10: Adversarial example generated with $\varepsilon = 1.5$.

lower tiles, we added a (1×1) tile-size for the inpainting GAN, that is, every pixel is assigned to a random mask.

The first experiment that we put in place consider only a single tile size. We do not aggregate the results by average but we directly predict the output of the GAN. We want to discover if a single GAN pass successfully removes the adversarial perturbation or if we would need more. To give an example of the different inpainting strategy, we will show the differences of each inpainting with respect to the Adversarial Example shown in Figure 5.10, generated with $\varepsilon = 1.5$.

Figure 5.11: Differences in the inpainting results by changing the tile size, for the single tile size version Massive Inpainting strategy.

As it can be seen in Figure 5.11, lower sized tiles better maintain the details of the original picture while increasing the pixel-level noise. Bigger tile size have intra-tile coherency and thus the noise is lower but many details are lost.

The results of the experiment shown in Table 5.4 confirm that the images inpainted with tile sizes bigger than (2×2) lose too much useful detail for the classification. Still, also bigger tile sizes have an higher accuracy than the

Attack	Adv Top-1	Tile Size (1, 1)	Tile Size (2,2)	Tile Size (4,4)	Tile Size (8,8)
Inception v3					
DeepFool	34.96%	**88.62%**	77.93%	57.45%	39.59%
FGSM	35.55%	**84.40%**	78.16%	56.96%	40.16%
PGD	3.61%	**88.01%**	77.34%	57.40%	39.53%
ResNet-101					
DeepFool	15.15%	**77.91%**	73.59%	54.07%	41.29%
FGSM	15.33%	70.09%	**73.17%**	53.06%	41.12%
PGD	1.57%	**76.57%**	73.64%	54.06%	0.00%
VGG16					
DeepFool	14.49%	54.22%	**58.55%**	37.95%	24.80%
FGSM	12.55%	40.90%	**58.01%**	35.86%	0.00%
PGD	3.95%	43.94%	**57.99%**	37.26%	24.84%

Table 5.4: Ablation tests for the single-size tile masking version of Massive Inpainting. This experiments run on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 1.5$.

adversarial top-1 that means that our defense is removing the adversarial noise, but it also need to maintain the details needed for the classification.

Interestingly, we see that yet by using a single tile size inpainting, our Defense by Massive Inpainting is able to recover up to 88% of accuracy from the adversarial examples. In general, the (1×1) tile size maintains more details and is the one with the highest performance in general, but on networks which suffers the presence of noise more than Inception v3, also a tile size of (2×2) performs well.

The natural question to study is now if the accuracy could benefit of a combination of the details of the (1×1) tile size with the lower noise of the (2×2) tile size, as in the central image of Figure 5.12. In other words, we see if by using a multi tile size inpainting with average aggregation as in the original RIAD proposition, the defense could benefit or not.

A detail of each of the three pictures of Figure 5.12 is shown in Figure 5.13. We immediately see that the version which aggregates by average the (1×1) version with the (2×2) version is more smooth and do not show as much noise as the single (1×1) version. At the same time, the details are well defined but obviously not as in the single (1×1) version. We would need to test such aggregation to see if the networks can accept some noise in

Figure 5.12: Differences in the inpainting results by changing the tile size, for the multi tile size version Massive Inpainting strategy.

Figure 5.13: Differences in the inpainting results by changing the tile size, for the multi tile size version Massive Inpainting strategy.

change of clearer details or if it would prefer a smooth version at the cost of fewer details.

The result of the multi tile size experiment are shown in Table 5.5. If we take into account the difference between the pictures in Figure 5.13, we see more clearly the difference between the three networks.

Inception v3, which can better support the noise added by the single version (1×1) tile size inpainting, still keeps preferring this single version, even if the results compared to the multi size (1×1) and $(2 \times 2)]$ does not show evident differences.

In ResNet-101 experiments, the network which can handle the noise added by our GAN better than VGG16 but still suffers if compared to Inception v3, we see that all the results start to indicate that the multi tile size $[(1 \times 1)$ and $(2 \times 2)]$ would provide better accuracy, even if also in this case the

Attack	Adv Top-1	Tile Size (1, 1)	Tile Sizes (1, 1), (2, 2)	Tile Sizes (1, 1), (2, 2), (4, 4), (8, 8)
Inception v3				
DeepFool	34.96%	**88.62%**	87.51%	83.45%
FGSM	35.55%	84.40%	**84.44%**	81.44%
PGD	3.61%	**88.01%**	87.37%	82.79%
ResNet-101				
DeepFool	15.15%	77.91%	**78.11%**	74.93%
FGSM	15.33%	70.09%	71.00%	**71.04%**
PGD	1.57%	76.57%	**77.01%**	74.68%
VGG16				
DeepFool	14.49%	54.22%	55.39%	**62.85%**
FGSM	12.55%	40.90%	43.16%	**47.67%**
PGD	3.95%	43.94%	46.19%	**51.59%**

Table 5.5: Ablation tests for the multi-size tile masking version of Massive Inpainting. This experiments run on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 1.5$.

differences are not high. As the compromise between those three networks is that the more fine-grained mask we use, the more details we get but also the more noise we get, we can confirm that ResNet does not achieve the same performance of Inception because it suffers more the noise produced by (1×1) inpainting mask.

Finally in the experiments performed with VGG16, we see that the best results are achieved with the network which mediates the (1×1) inpainting outcome with the outcome from the (2×2), (4×4) and (8×8) masks. Such strategy of using 4 different tile sizes produces very little noise at the cost of having more blurred details if compared to single small tile size inpainting or to dual small size inpainting as $[(1 \times 1)$ and $(2 \times 2)]$. In case of VGG16 we proved that the noise that the GAN adds at $\varepsilon = 1.5$ is too much for the classifier and thus it prefers the full $[(1 \times 1), (2 \times 2), (4 \times 4), (8 \times 8)]$ inpainting at the cost of a decreased performance.

5.3.2 *Comparison with other defense*

Once identified the best hyper-parameters for our proposed defence, we are ready to compare its performance with our baseline and our Saliency Based

method to highlight the improvements and the motivation behind them. Finally, we will measure the capabilities of our proposed method without any attack, because the defense should not impact too much on clean accuracy, and we will make some final considerations about different experiments in a broad ε range.

So far, we identified the following method: Massive DeepFill v1 Inpainting via (1×1) RIAD-like tiles and random masking of $1/12$ of the image at a time, not followed by any denoiser and submitted to an Inception v3 classifier.

5.3.2.1 *Massive Inpainting and Saliency-Based Inpainting*

The first comparison we performed is with respect to our saliency based localization and inpainting strategy that we described in Chapter 3. This latter architecture was able to overtake our previous baseline of a few percent points while increasing of three times its speed.

Still, the Saliency Based localization and Partitioned Inpainting strategy requires more than 16 GAN runs in addition to the time required by the saliency localization strategy. As our Massive Inpainting approach does not have the localization stage and it needs only 12 GAN runs, we reached another time improvement reducing the required time of the defense to just 400 milliseconds, including the classification prediction time, resulting in a very small overhead.

In addition to the speed improvement we have also reached a tangible accuracy improvement, as it can be seen in Table 5.6. Every test executed on each network and attack reported an higher accuracy using our Massive Inpainting strategy with respect to the Saliency based one. This confirms our initial guess about trust in presence of adversarial examples: the lower it is, the better it is. Having successfully removed all the trust from the input image by massively inpainting all and each pixel, is showing a final top-1 accuracy which is higher than structured approaches such as Saliency Based Localization and Partitioned Inpainting which still trust some areas of the image.

It is worth to notice that it may happen that the inpainted image which is sent to the classification network in our Massive Inpainting strategy could be perceived as worse than the one of Saliency Based by a human person, but for sure our inpainted result does not contain any adversarial feature and thus any misprediction is to be imputed to the added noise of the method and the noise robustness of the classification model.

To better document our statement about the fact that the low accuracy performance of Saliency Based Localization and Inpainting is due to its trust and not to the method, we run an experiment in which we localize the salient object with Saliency Based Localization and then we denoise the foreground and inpaint the background with RIAD-like masking and not anymore with

Attack	Adv Top-1	Adv Top-5	Massive Proposed	Saliency Best
Inception v3				
DeepFool	34.99%	78.18%	**89.43%**	76.72%
FGSM	35.55%	67.79%	**89.60%**	77.22%
PGD	3.61%	11.86%	**88.73%**	72.16%
ResNet-101				
DeepFool	15.15%	68.63%	**83.51%**	65.13%
FGSM	15.33%	55.21%	**83.24%**	65.60%
PGD	1.57%	30.33%	**83.03%**	60.07%
VGG16				
DeepFool	14.49%	69.56%	**67.97%**	55.46%
FGSM	12.55%	47.33%	**68.03%**	56.04%
PGD	3.95%	36.14%	**68.00%**	53.05%

Table 5.6: Comparison between our proposed Massive Inpainting method and the Saliency Based Localization and Partitioned Inpainting strategy, tested on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$.

Partitioned Inpainting. We then compare this result with its opposite which uses the denoiser on the background and the RIAD-like inpainting strategy on the foreground, and we compare it to the Massive Inpainting we are proposing.

Attack	Adv Top-1	Adv Top-5	Saliency + RIAD BG	Saliency + RIAD FG	Massive Inpainting
Inception v3					
DeepFool	34.96%	78.19%	76.02%	76.00%	**88.62%**
FGSM	35.55%	67.79%	65.46%	65.16%	**84.40%**
PGD	3.61%	11.86%	63.74%	63.54%	**88.01%**

Table 5.7: Comparison between a Massive Inpainting Strategy and a Saliency Based strategy whose inpainting technique is the same used for our Massive Inpainting architecture. Experiments are run over the Inception v3 classifier architecture with a value of $\varepsilon = 1.5$.

The results of this experiment are shown in Table 5.7. We see that nor inpainting only the background while denoising the foreground, nor inpainting only the foreground while denoising the background, has a top-1 accuracy

comparable with our proposed Massive Inpainting strategy. As the only thing which changes between Saliency with RIAD-like inpainting and Massive Inpainting is just the trust posed in the image (we use the same inpainting strategy), we see that our approach of switching to a zero-trust adversarial defense was indeed correct.

5.3.2.2 *Comparison with our baseline*

Having verified that the Defense by Massive Inpainting is our best model with respect to all the other proposed approaches of Chapter 4, we can show the improvements compared with our baseline CIIDefence [1]. We run our experiments using two different ε values to better assess possible variations of the method linked to the increase of ε.

	Eps = 0.2			Eps = 2.0		
Attack	Adv Top-1	Massive Proposed	Baseline Best	Adv Top-1	Massive Proposed	Baseline Best
Inception v3						
DeepFool	48.47%	**89.43%**	75.87%	34.99%	**88.03%**	67.35%
FGSM	50.62%	**89.60%**	76.56%	35.55%	**85.81%**	53.81%
PGD	35.78%	**88.73%**	70.22%	3.61%	**87.88%**	43.07%
ResNet-101						
DeepFool	40.14%	**83.51%**	63.50%	15.15%	**75.71%**	37.57%
FGSM	41.39%	**83.24%**	64.26%	15.33%	**64.72%**	31.18%
PGD	30.59%	**83.03%**	58.72%	1.57%	**75.57%**	16.44%
VGG16						
DeepFool	45.89%	**67.97%**	55.58%	14.49%	**47.97%**	27.49%
FGSM	47.22%	**68.03%**	55.82%	12.55%	**31.27%**	20.79%
PGD	42.67%	**68.00%**	53.09%	3.95%	**36.34%**	10.63%

Table 5.8: Comparison between our proposed Massive Inpainting method and our Baseline CIIDefence [1] tested on the Inception v3, ResNet-101 and VGG16 classifier architectures with a value of $\varepsilon = 0.2$ and $\varepsilon = 2$.

Unsurprisingly, in Table 5.8 we see that our method overtakes our baseline both at low epsilon values and at high epsilon values. In addition we notice again the stability that our method has between different epsilons: using Inception v3 as classifier, the mean accuracy drop of switching from $\varepsilon = 0.2$ to $\varepsilon = 2$ is just 2%. At the opposite, our baseline drops over 20% when ε changes from $\varepsilon = 0.2$ to $\varepsilon = 2$.

This stability of the results over such range of ε is a feature of our proposed defense and will be further investigated to assess its potentialities and limits.

5.3.3 *Consistency of the results*

As just highlighted, our proposed defense strategy has stable results over both small and high ε values. However, to understand our defense consistency, we need to investigate how and where the performance start to drop. Indeed, we recall that the formalization of the concept of Adversarial Attack does not define a maximum ε but just impose that the adversarial perturbations should be "small enough" to avoid being recognized by a human as altered images.

By keeping increasing ε, sooner or later the network will surrender, with the hard limit for any defense that is $\varepsilon = 128$ over 255. By allowing such perturbation indeed the attacker could simply set each pixel to be equal to 128 (subtracting up to 128 to light pixels, adding up to 128 to dark ones) and generate a totally grey image which can not be classified. However, these value of ε do not have any importance as over $\varepsilon = 2$ some human could start to spot adversarial patterns and nevertheless in the literature the higher tests have $\varepsilon = 8$, even if with such important variation the attack is for sure spotted by humans.

5.3.3.1 *performance for a fixed epsilon range*

In the following, we will only test the Inception v3 classifier to assess the top-1 accuracy of our proposed Massive Inpainting strategy as it is the network which better handle the noise generated by the (1×1) inpainting performed by our method.

The results are shown in Table 5.9. We can see that as epsilon increases, the adversarial top-1 and top-5 move towards 0%, and also our defended top-1 accuracy decreases. However, our defense as epsilon doubles loses only up to 1% accuracy with DeepFool and loses only up to 2% with PGD. The story is different for FGSM as our defense still achieve good performance but the accuracy drops over 10% with respect to the previous test with halved epsilon.

The motivation for this result can be found on how the two different attacks work: as explained in Section 2.2.1, PGD is a more powerful attack which can exploit at each iteration the correct direction of the gradient to generate Adversarial Examples which are composed of the minimal variation needed to be predicted in a wrong class. On the contrary, FGSM is a computed one-shot method which approximate the nearest boundary direction with the gradient direction at the first computation, thus the Adversarial Examples generated by FGSM have more noise compared with the other attacks.

Attack	Adv Top-1	Adv Top-5	DF1 Massive Inpainting
Inception v3 @ Eps = 2			
DeepFool	33.24%	77.14%	88.03%
FGSM	35.05%	67.62%	85.81%
PGD	2.00%	8.66%	87.88%
Inception v3 @ Eps = 4			
DeepFool	28.80%	75.31%	87.79%
FGSM	31.20%	61.06%	73.23%
PGD	0.27%	2.19%	86.36%
Inception v3 @ Eps = 8			
DeepFool	27.02%	74.44%	87.07%
FGSM	30.92%	60.98%	63.48%
PGD	0.08%	0.77%	83.27%

Table 5.9: Attacks on our proposed Massive Inpainting methods with ε that ranges from 2 to 8. Those attacks are tested only on the Inception v3 architectures for better comparison.

Figure 5.14: Comparison of FGSM and PGD at $\varepsilon = 2$. The two image can be perceived as similar and it is difficult to spot the presence of the adversarial perturbation.

As we can see in Figure 5.14, with a perturbation $\varepsilon = 2$ the two adversarial examples can be easily confused with the original real images, even if the attacks manage to drop the accuracy from 100% to 33% and 2% respectively.

Figure 5.15: Comparison of FGSM and PGD at $\varepsilon = 4$. While it is easy to spot some perturbation in the FGSM generated Adversarial Example, PGD still manages to keep the adversarial noise well hidden.

It is important to note that from those values, our defense is able to restore an accuracy of over 85% and 88% for the two attacks, thanks to the fact that the adversarial noise is limited.

As ε increases up to 4, we can see in Figure 5.15 that while the presence of PGD is still well hidden in the picture (even if the adversarial accuracy is only 0.27%), FGSM has added a huge amount of noise and we can now easily spot the presence of such attack. Even if the added noise is still in the L_∞ bound, the presence of such perturbation act as noise for our GAN and thus explained the performance drop if compared to the others.

Finally, if we inspect the images generated with $\varepsilon = 8$ as in Figure 5.16, we immediately see that PGD starts to be noticed, while FGSM is gone far out of the boundaries of the "small enough".

We do not put too many attention on the results obtained by our defense if applied to images that can not be considered as Adversarial Examples, as they lacks the fundamental propriety: the slightly perturbation. Instead, we want to keep studying the performance of our defence in presence of PGD and DeepFool, the only two attacks which still manage to create altered inputs which can not be clearly distinguished from clean inputs.

Analyzing the results of DeepFool, we see that from $\varepsilon = 2$ to $\varepsilon = 8$ our Massive Inpainting defense loses only less than 1% of accuracy, while the attack successfully achieve a reduction of the performance of the undefended network of over 6%. Here, we are more interested in the stability of the results rather than the single experiments outcomes, as following the guidelines of [5], there is no single epsilon values which can measure the power of the

FGM with epsilon = 8

PGD with epsilon = 8

Figure 5.16: Comparison of FGSM and PGD at $\varepsilon = 8$. FGSM has totally compromised the image while PGD starts to be noticed.

defence: attacks are various and different one another and valuing consistency it is more important than single results.

Looking at PGD, it does not have a huge drop in the undefended accuracy as DeepFool because the accuracy is already low, that is 2% yet at $\varepsilon = 2$. Increasing ε gives more power to PGD of creating adversarial examples also for images which, thanks to their nature, are difficult to mispredict with only small perturbations, but does not change substantially other images. We consider the 0.08% accuracy at $\varepsilon = 8$ to be a casual noise: these 8 images over 10000 can not be easily misclassified by means of PGD. Indeed, as it can be said in Table 5.10 even with higher epsilons the accuracy maintains itself over 0.00%.

The results obtained with higher ε showed in Table 5.10 outline the stability of our method once more. By giving the ability to the attack to change each channel of each pixel up to ± 20 units, the Massive Inpainting is able to reconstruct images which can still be classified correctly in over 85% of times in presence of DeepFool and over 70% of times in presence of PGD.

To give an idea of the magnitude of the perturbation, we can see in Figure 5.17 an example image drawn from the perturbed images generated by DeepFool. We can see that the attack ha almost completely modified the shape of the snout and the ears of the quadruped making it hard also for a human to recognize the correct class from the different quadruped class contained in ImageNet.

As ε reaches 20, it would not make sense to continue our analysis even more. The amount of noise added is starting to make difficult to recognize the true class also by a human and as in can be seen in Figure 5.18, also the Massive

Attack	Adv Top-1	Adv Top-5	DF1 Massive Inpainting
Inception v3 @ Eps = 12			
DeepFool	26.93%	74.45%	86.25%
FGSM	33.63%	62.87%	59.45%
PGD	0.08%	0.50%	79.77%
Inception v3 @ Eps = 16			
DeepFool	26.81%	74.42%	85.94%
FGSM	35.29%	63.92%	57.04%
PGD	0.09%	0.43%	76.02%
Inception v3 @ Eps = 20			
DeepFool	26.46%	74.34%	85.02%
FGSM	36.76%	64.66%	51.81%
PGD	0.08%	0.31%	70.13%

Table 5.10: Attacks on our proposed Massive Inpainting methods with ε that ranges from 12 to 20. Those attacks are tested only on the Inception v3 architectures for better comparison.

Inpainting GAN start to suffer the presence of such noise. Nevertheless, the defense strategy is still increasing the prediction accuracy also in presence of FGSM with $\varepsilon = 20$ from 36% to 52%.

Notice that with attacks which keep the unnecessary noise limited as DeepFool, our proposed defense is able to maintain consistent its performance with a defended top-1 accuracy of DeepFool that goes from 88% at $\varepsilon = 2$ to a top-1 accuracy of 86% at $\varepsilon = 16$. Similarly with PGD, Massive Inpainting maintains consistent its performance with a defended top-1 accuracy that goes from 88% at $\varepsilon = 2$ to a top-1 accuracy of 76% at $\varepsilon = 16$.

5.3.3.2 *performance with no attack in place*

Taking inspiration from the requirements of a defense method exposed in [4], until now we have verified that our defense method:

- Has minimal impact on the architecture
- Maintains speed of network
- Works for adversarial samples relatively close to points in the training dataset

Figure 5.17: Example of an image generated by DeepFool with $\varepsilon = 20$

We need to prove that our defense strategy also maintains as much as possible the original accuracy of the network when no attack is put in place.

Attack	Adv Top-1	Adv Top-5	DF1 Massive Inpainting	DF2 Massive Inpainting
Inception v3				
No Attack	100.00%	100.00%	92.16%	96.33%
ResNet-101				
No Attack	100.00%	100.00%	90.33%	95.92%
VGG16				
No Attack	100.00%	100.00%	84.79%	93.09%

Table 5.11: Accuracy measured on clean images tested on Massive Inpainting using the Inception v3, ResNet-101 and VGG16 classifier architectures. In this case $\varepsilon = 0$ because no attack is put in place.

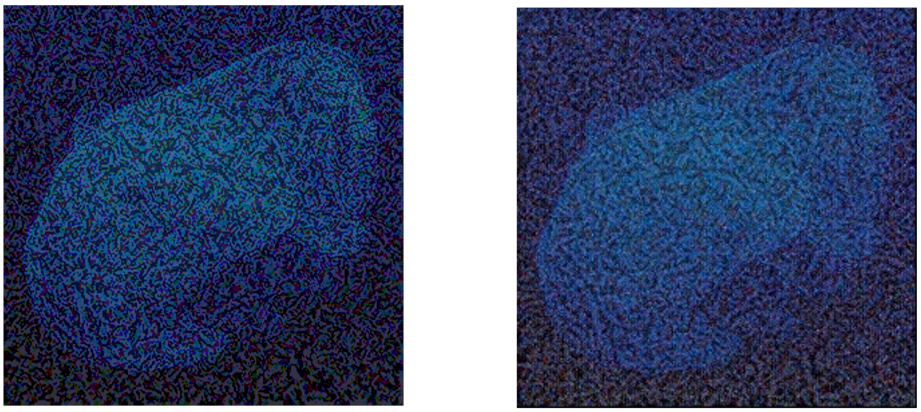


Figure 5.18: Example of an image of a jellyfish generated by FGSM with $\varepsilon = 20$ along with its inpainted version.

In Table 5.11 we can see how Massive Inpainting performs on clean inputs. The architecture is able to keep the accuracy always 90% and here we see a difference between the usage of DeepFill v1 and DeepFill v2. While until now we have executed our test especially using DeepFill v1, we note that on clean inputs the partial convolutions used in DeepFill v2 helped to preserve the fine-grained details. As stated in Section 5.3.1.2 however, DeepFill v2 is more vulnerable to the presence of noise in the image, motivation behind the usage of the first version of the GAN.

We found DeepFill v1 to achieve lower performance in average, but stable. On the contrary, DeepFill v2 achieves the best performance on clean input but rapidly decrease its accuracy in presence of noise or adversarial perturbations.

5.4 REMARK

In Section 5.1.2 we hypothesized that the functioning of Massive Inpainting is tightly related to the transformation of the adversarial perturbation in noise. More specifically we believe that the Generative Adversarial Network which performs the inpainting tries to infer the masked pixels thanks to the correlation that pixel has with its neighbors. When Adversarial Examples are within a certain perturbation bound that depends on the attack and that we identified to be $\varepsilon \in [0, 16]$ for DeepFill and PGD and $\varepsilon \in [0, 8]$ for FGSM (due to the added noise), the GAN successfully manages to infer pixels based on their mutual correlation linked to the true class.

As the experiments confirms, our proposed defense we were able to alter the microscopic level value of each pixel via (1×1) inpaintings in such a way that the adversarial features at macroscopic level are broken as described in Section 5, and the misprediction rate is mostly due to noise.

However, as it can be seen in Figure 5.18, when we increase the perturbation over a certain threshold and the real figure is not well defined, the Generative Adversarial Network starts to infer missing pixels based on the mutual correlation with the adversarial perturbation itself. In this way, following a zero-trust approach and inpainting all the pixel the defense can still increase the accuracy but at the same time it can not manage to fully erase the adversarial perturbation.

In general, even with acceptable ε, we found many inpainted images to be less visually pleasant to a human with respect to the adversarial example from which they are generated. However, the adversarial examples have low accuracy while our inpainting outcome recovers a good accuracy even if those image are less human-pleasant. The phenomena could be due to the transformation of the hidden adversarial perturbation in clear noise in the image. To validate such hypothesis we set up a final analysis based on the Structural Similarity Index Method [53] (SSIM) and performed on our whole test dataset (10000 images). We have chosen SSIM with respect to other image quality assessments like RMSE because SSIM is a perception based model which emphasizes the strongly inter-dependant pixels or spatially closed pixels. By definition, SSIM measures the change of perception in structural information (which is what we are interested in), while RMSE measure just the absolute change of pixel values.

1. We compute the SSIM of all our adversarial examples with $\varepsilon = 16$ with respect to the original image, and we find this value to be equal to $SSIM = 78.45\%$.

2. Then, we compute the Structural Similarity Index of all our inpainted results with respect to the original image before the adversarial attack (clean input). We find this value to be $SSIM = 59.33\%$

3. Finally, we compute the Structural Similarity Index of the clean image perturbed with some Salt and Pepper noise modified to comply with the L_∞ norm equal to 16. We subtract 16 to pepper pixels and add 16 to salt pixel. A comparison of adversarial image and S&P image in shown in Figure 5.19. We compute the SSIM index of this noised image with the clean input over all our dataset and we find it to be $SSIM = 57.43\%$

This simple experiment which results are summarized in Table 5.12 proved that in presence of adversarial attack there is no correspondence between the SSIM index and the quality of an image measured as the ability for a classifier to correctly recognize its true class. Indeed, our inpainted input is

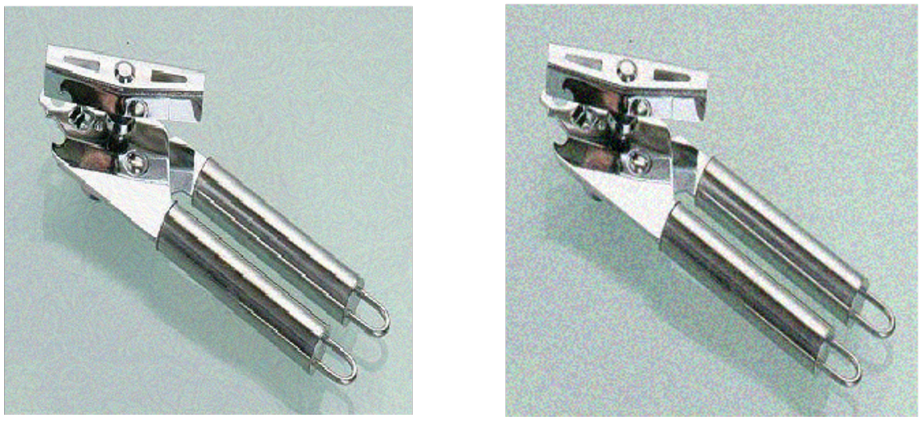


Figure 5.19: Example of an image of a jellyfish generated by FGSM with $\varepsilon = 20$ along with its inpainted version.

less similar to the clean input but has a very higher accuracy if compared to the adversarial image.

SSIM	Clean Image	Adv Image	Inpainted Image	S&P Image
Clean Image	100.00%	78.45%	59.33%	57.43%

Table 5.12: Summary of the results of the noise test. It can be seen that the adversarial image is more similar to the clean image with respect to the inpainted outcome, even if the latter accuracy is higher.

In particular, we find that the SSIM measure of the inpainted input with respect to the clean image is not bigger than the adversarial input one. On the contrary, the inpainted image is more similar to an artificially noised image with same epsilon as the attack, than to the original input. This fact could be explained by taking into consideration the de-correlation operation performed by adversarial attacks and the reconstruction operation just described performed by the GAN.

6 CONCLUSION

We've come a long way towards understanding adversarial robustness.
We still have a long way to go.

— Carlini, *Adversarially non-Robust Machine Learning*, 2021

Thanks to the scientific literature of the past few years it is nowadays well known that Neural Networks models are vulnerable to adversarial examples: artificially perturbed inputs almost indistinguishable from natural data that are classified incorrectly by the network. Some of the latest findings in addition suggest that the presence of adversarial attacks may be an inherent propriety of deep learning models. This phenomenon is a threat to the deployments of such models in real-word scenarios, especially if these domains are safety critical. Even if approximately each year a new accuracy improvement appears for standard deep learning models which operates without considering adversarial attacks, such intrinsic weakness has not yet been solved.

To address this problem, which is critical for the mass-adoption of deep learning models, we studied the adversarial robustness of neural networks through the lens of a zero-trust approach. Such reasoning has provided us with a broad view on much of the prior work on this topic and was conjugated by us as a trust-recovering problem through image inpainting because our problem was related to the computer vision world, but it can be easily extended on other domains such as audio and speech recognition through audio wave reconstruction.

This approach has always taken into consideration real-world use cases and that was the reason why we did not try a robust learning approach: in some domains it may be unfeasible to perform a full retrain of a critical deep learning model just to increase its accuracy over perturbed data, because the retrain takes a lot of time and if not carefully planned and designed it may lead to a degradation of clean performance. The high difficulty of this approach is confirmed by the high number of research works which apply variations of robust learning to small datasets like MNIST of CIFAR ones, but only a very limited number extend it to complex dataset like ImageNet.

In addition to this, in all our study we also have followed advises from influent scientists which have brought to the adversarial attacks literature many new discoveries. Some of them, such as *Anish Athalye, Nicholas Carlini, David Wagner, Ian Goodfellow, Nicolas Papernot, et al.* often state best practices to make research works on this topic reproducible and useful for the scientific

community. We tried to apply all their advises in our work; the most challenging best practice to follow was related to the full white-box evaluation as described in [5], where no detail is left to chance in order to correctly define a defense strategy as "fully white box", but many other such as the public availability of the code were also followed.

It is worth mentioning that the accuracy of our proposed defense method could be improved even more by increasing the expressiveness and generative power of the GAN. Training GANs is a challenging task and an active area of research, and the performance of Defense by Massive Inpainting are highly related to a good inpainting quality. In addition, our defense has only one hyperparameter, the size of the tiles of the mask for the GAN. Choosing this hyperparameter is not a difficult task and this makes Defense by Massive Inpainting an easier to implement defense.

While working on our baseline we faced many problems related to the adversarial world, like performing white-box attacks, analyzing the robustness of a classifier, and understanding the basis behind how to make a defense from attacks without altering too much the clean accuracy of the network. Finally, we needed a way to choose which attacks could we use for the result comparison as the real goal is not to defend from specific attacks, but to defend also from unseen attacks. We have chosen the most used ones in the literature, adding also some variance: we took PGD because is one of the strongest attacks used nowadays, we took FGSM which is an old and well know attack but we also took DeepFool which is a newly-presented fast attack with low added perturbations. We believe that robustness against such well-defined classes of adversaries is an important step towards fully resistant deep learning models.

Our main discoveries are the following:

- **Strategies which trust less the input are more robust than models which trust it more.** Our Defense by Massive Inpainting (which does not trust the input at all) has proven to be more accurate with respect to both our baseline (which trusts a lot the input) and with respect to Saliency Based Localization and Inpainting (which trusts less the input). This is explained by the fact that in our threat model the adversary has full power of perturbing the input in the strongest possible way, and if we tell to our adversary how the defense work (as in a full white-box scenario) then he would be able to compute which parts of the image do we trust and he may use this information to strengthen its attack.

- **Simpler defenses can be as powerful as complex ones.** Our Defense by Massive Inpainting has many similarities with both our baseline and our Saliency Based Localization and Inpainting, but it totally removes an entire pipeline stage: the localization one. What our Massive Inpainting is doing is just inpainting the image in a smart way, without the need

of other complex computations to identify differences of importance in the image. This allows a time reduction of up to 75% and an increase in the accuracy of up to 50%.

- **Plug and Play defenses are not the only solution.** All our work is based on Plug and Play solutions. Someone who needs to secure its model can take our proposed defense, place it in front of its model and use it as it normally does. However, this would be useful if the domain requires it, but may be suboptimal if its domain does not have too strict constraint on the retraining. In such case, it is also possible to choose to follow a robust learning approach. In other circumstances, even if the retraining may be possible, it is more convenient to opt in for our Plug and Play defense. In summary, there is not yet a defense which can eliminate the problem of Adversarial Examples at all, but we think that the results of our Thesis can be useful for the future research about this topic.

- **Trade-offs are the normality.** Trade-offs are common in every aspect of Computer Science and Adversarial Attacks of Deep Learning models are not an exception. We discovered a trade-off between the time required by the defense and the accuracy, and another one more linked to the inpainting between the quality of the inpainting and the presence of noise. In general, these trade-offs must be manually solved with respect to the different domains in which we are employing the defense and their relative constraint (i.e. on time, on the classifier).

- **Inpainting could help in transforming the adversarial perturbation in noise.** We discovered that adversarial examples are way more similar to the original clean image than inpainting results if analyzed trough the Structural Similarity Index Method (SSIM). At the same time, inpainted results seem to be similar to clean inputs as much as a random Salt and Pepper noise generated by limiting the L_∞ norm to the same value of the adversarial one. These results can be explained taking into account that the GAN that we use tries to infer pixel values based on the learnt correlation that there exists between neighboring pixels in natural images. As adversarial examples does not exhibit this correlation because they are altered, non-natural images, the adversarial perturbation is converted into noise. Then, if we submit the noisy inpainting result to a network which handles well the noise the accuracy is almost recovered, otherwise we need to take actions to remove some of the GAN generated noise by means of denoising filters or multiple size inpaintings, but doing so will destroy part of the details of the image.

- **Inpainting could protect from full-white box threats.** Many accepted papers in the literature present white box techniques without a careful

assessment of the added attack surfaces that the proposed defence brought in. Many of such defenses could not be broken via standard attacks targeting the classification network but easily fail with attacks targeting the defense. This is demonstrated by the huge amount of scientific papers which are published each year to disprove previous results. The inpainting however does not just introduce a non-linearity as other gradient obfuscation based defense do, but instead it totally changes the input based on mutual learnt natural correlation. Integrating such technique in a defense could help to protect from full-white box attacks because at the moment the only known way to make the inpainting GAN to be bad for a classification task is by increasing the perturbation, a thing that is allowed only under a constraint in adversarial examples.

6.1 FUTURE WORKS

While developing this Thesis along with the two proposed framework Saliency Based and Massive Inpainting we followed a specific path that led us in finding the robust defense which we called Defense by Massive Inpainting. However, there are other possible options risen during the work which were not extensively explored to help keeping the focus on what we thought was more important. These option can be the source of new interesting insights about adversarial attacks and in general about learning algorithms. The following are some of them:

- Better understanding of the difference in the accuracy between Inception v3, ResNet-101 and VGG16. Throughout all our Thesis, our proposed defense worked at its best with Inception v3, followed by ResNet-101 and only after by VGG16. We tried to give an explanation related to the maximum supported noise but more extensive tests could be performed to stress this point.

- Starting from the previous point, it is possible to study if the Massive Inpainting strategy could be better suited for certain network architectures or it does apply to all. If the Defense by Massive Inpainting is supported only by a subset of networks (maybe the ones which better handle the noise in the image), it should be identified in order to increase the accuracy and in order to have a deeper view of the defense's actions.

- Throughout our Thesis we focused only on L_∞ attacks. To better explore all the possible research directions however, studies about the effects of Massive Inpainting on L_2 Adversarial Examples should be done. In L_2 attacks the perturbation is more hidden because it can not have high single amplitude, but it can be more pervasive. We think that a

lower maximum amplitude (although followed by a broader number of impacted pixels) could suffer more the effect of the inpainting and thus our defense can be even more effective but this is an open point for a future work.

- This Thesis focuses on attacks against digital images, but physical attacks [55, 54] are attracting day after day more attention from the scientific community. In particular, patch-generated Adversarial Examples (widely used in physical attacks applied over objects) are out of the scope of this work. However, it would be interesting to study the effectiveness of our digital Defense by Massive Inpainting on physical attacks.

In addition, starting from our results, it is possible to research about specific improvements around our Massive Inpainting strategy:

- Incorporate our Defense by Massive Inpainting in a detector of attacks which chooses which images to predict between the original input and the inpainted image based on the difference between the two, similarly to what RIAD does.

- Considered that we found that smaller tile sizes better influence the reconstruction accuracy of the GAN for what concern the prediction task (while increasing the noise), we ended up using the smallest possible tile size. We wonder if and how an even smaller tile size could influence the defense. For example, it would be possible to consider one channel at a time, with a 3D mask, an inpaint channel-level values in the same way we are doing with Defense by Massive Inpainting. This would just require a slightly modification to the GAN but it could be justified by improved accuracy.

- Finally, we wonder about the possibility of training a Generative Adversarial Network with a loss which accounts for the classification accuracy of the classifier. Actually, we are using a GAN which was trained for reconstruction tasks and this fact has the obvious benefit of potentially apply the GAN to any network. However, if it would be possible to train the GAN already attached to a specific network which exposes the gradient (like Inception v3 and many other networks do) then the GAN will be bounded to the classifier but the performance may increase significantly because the GAN would have a clear and well defined gradient to follow in the training phase, while the training of the GAN with reconstruction losses often take into account too many factors to make human-pleasant images. This will be a difficult task because the classifier must be already trained and some considerations may be done about the possibility of fine-tuning it with the GAN but the benefits may be huge.

6.2 REFLECTION ON THE LONG TERM DIRECTION

Since 2015, the research on adversarial attacks has seen huge improvements.

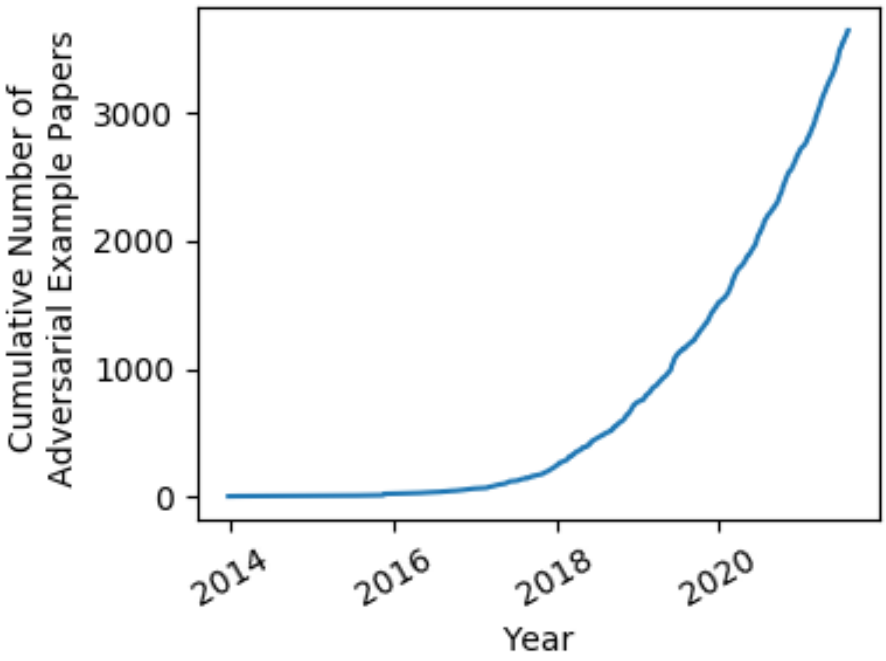


Figure 6.1: Cumulative number of paper regarding adversarial attacks published since 2014 taken from [63]

Nowadays, as shown in Figure 6.1, it is impossible to keep up to date with all the latest paper and research work in this field, because each day there are almost ten different publications studying new attacks or new defenses [63].

Even worst, the situation is becoming stuck in a lack of novelty. A couple of years ago all the papers claiming excellent performance against adversarial attacks and proposing new defenses techniques were broken thanks to smart ideas of some scientific researcher around the world (i.e. [19]). This process is an important part of the scientific learning process because by providing the ideas behind the new attacks which broke the proposed defences, the research knowledge increases.

However, nowadays the number of proposed defenses has reached an exponential growth and while some of these new works are doing shenanigans with the results to prove statements that are not always true or at least that are not as interesting as they are described, the bad part is that even honest works about new defenses were broken using state-of-the-art attacks (i.e. [15, 8]). According to Figure 6.2, new papers are not injecting new knowledge into the scientific community and also disproving papers are being rejected from

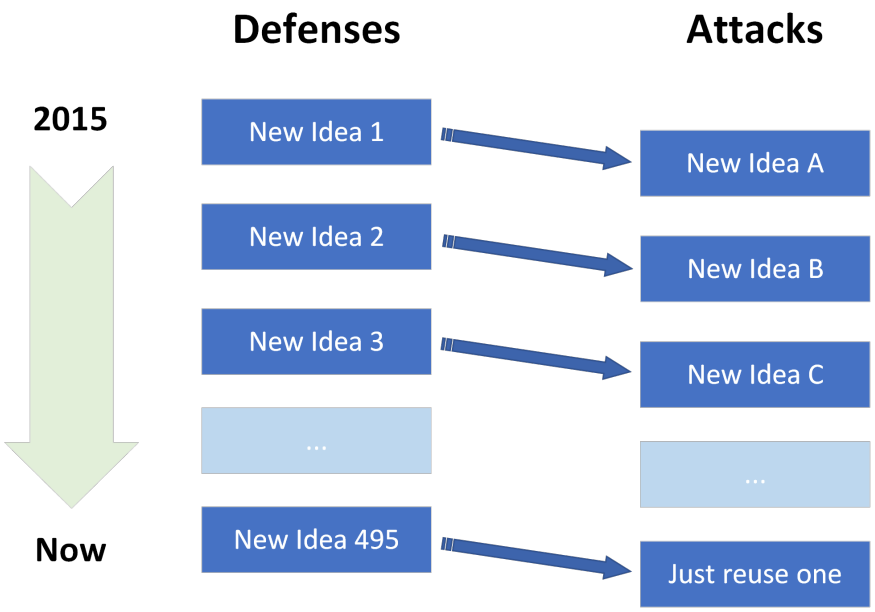


Figure 6.2: Graphical illustration of the no-novelty conjecture.

publication because they do not use new attacks to broke novel defenses, but they just adjust the parameters of well-known methods like PGD or BPDA.

Our personal opinion about the adversarial attacks problem of the Neural Networks is that it is just one of the problem that the scientific community has faced and will continue to face. For example, twenty years ago computer scientists all around the world were proposing new cryptographic standards. Just in 1997, Wagner co-authored more than 10 papers disproving previous results (i.e. [9, 12, 13, 10]). However nowadays there exists one cryptographic standard which has not been broken for over 20 years (AES [11]).

As Carlini states in [62], the adversarial attacks field can be considered at the same innovation level of cryptoanalysis pre-Shannon theorem. Before the Shannon theorem, researchers were proposing new cryptographic system just based out of empirical results: just like adversarial attacks nowadays. With the advent of such theorem and its strong implications, the methodological aspect of cryptoanalysis became stronger and that was the basis for such big improvements.

Like the introduction of Convolutional Layers in 1998 [14] totally changed the way in which image classifiers works, we think that the high-interest moment that adversarial attacks is experiencing (shown in Figure 6.1) should be exploited to give a new boost to the scientific literature with a clear and rigorous methodology on which researchers can start building on.

With this Thesis we claim to have found a defense strategy which can not be simply defeated to 0% by already existing attacks and thus we hope to give a new content to the scientific research. We are perfectly aware that there will require years and hard work before finding a definitive solution to adversarial attacks, but we consider this novel defense strategy a good advance in this direction.

BIBLIOGRAPHY

[1] Puneet Gupta and Esa Rahtu. "CIIDefence: Defeating Adversarial Attacks by Fusing Class-Specific Image Inpainting and Image Denoising." In: Oct. 2019, pp. 6707–6716 (cit. on pp. 9, 17, 23–25, 32, 35, 37, 39, 53, 54, 66, 89).

[2] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. *Practical Black-Box Attacks against Machine Learning*. 2017. arXiv: 1602.02697 [cs.CR] (cit. on p. 21).

[3] Aaditya Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James Storer. *Deflecting Adversarial Attacks with Pixel Deflection*. 2018. arXiv: 1801.08926 [cs.CV] (cit. on pp. 17, 25, 66).

[4] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. *Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks*. 2016. arXiv: 1511.04508 [cs.CR] (cit. on pp. 20, 94).

[5] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. *On Evaluating Adversarial Robustness*. 2019. arXiv: 1902.06705 [cs.LG] (cit. on pp. 17, 25, 34, 49, 67, 75, 76, 92, 100).

[6] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. *Object Detectors Emerge in Deep Scene CNNs*. 2015. arXiv: 1412.6856 [cs.CV] (cit. on p. 26).

[7] Alexei A. Efros and William T. Freeman. "Image Quilting for Texture Synthesis and Transfer." In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. New York, NY, USA: Association for Computing Machinery, 2001, 341–346. ISBN: 158113374X (cit. on p. 27).

[8] Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. *On Adaptive Attacks to Adversarial Example Defenses*. 2020. arXiv: 2002.08347 [cs.LG] (cit. on p. 104).

[9] Chris Hall, John Kelsey, Vincent Rijmen, Bruce Schneier, and David Wagner. "Cryptanalysis of SPEED." In: *Selected Areas in Cryptography*. Ed. by Stafford Tavares and Henk Meijer. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 319–338. ISBN: 978-3-540-48892-7 (cit. on p. 105).

[10] David Wagner. "Slide Attacks." In: *Fast Software Encryption*. Ed. by Lars Knudsen. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 245–259. ISBN: 978-3-540-48519-3 (cit. on p. 105).

[11] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. "Biclique Cryptanalysis of the Full AES." In: *Advances in Cryptology – ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 344–371. ISBN: 978-3-642-25385-0 (cit. on p. 105).

[12] D. Wagner, N. Ferguson, and B. Schneier. "Cryptanalysis of FROG." In: 1998 (cit. on p. 105).

[13] David A. Wagner, L. Simpson, E. Dawson, J. Kelsey, W. Millan, and B. Schneier. "Cryptanalysis of ORYX." In: *Selected Areas in Cryptography*. 1998 (cit. on p. 105).

[14] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cit. on p. 105).

[15] Anish Athalye and Nicholas Carlini. *On the Robustness of the CVPR 2018 White-Box Adversarial Example Defenses*. 2018. arXiv: 1804.03286 [cs.CV] (cit. on p. 104).

[16] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. "Learning Deep Features for Discriminative Localization." In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2921–2929 (cit. on p. 26).

[17] Kerckhoffs Auguste. *La cryptographie militaire*. 1883. Journal des sciences militaires: 9:538 (cs.LG) (cit. on p. 17).

[18] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. *Image Super-Resolution Using Deep Convolutional Networks*. 2015. arXiv: 1501.00092 [cs.CV] (cit. on p. 23).

[19] Anish Athalye, Nicholas Carlini, and David Wagner. *Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples*. 2018. arXiv: 1802.00420 [cs.LG] (cit. on pp. 21, 34, 49, 61, 76, 77, 104).

[20] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. *Adversarial examples in the physical world*. 2017. arXiv: 1607.02533 [cs.CV] (cit. on p. 11).

[21] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. *Do Adversarially Robust ImageNet Models Transfer Better?* 2020. arXiv: 2007.08489 [cs.CV] (cit. on p. 22).

[22] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images." In: (2009) (cit. on p. 22).

[23] Eero P. Simoncelli. "Bayesian Denoising of Visual Images in the Wavelet Domain." In: *Bayesian Inference in Wavelet-Based Models*. Ed. by Peter Müller and Brani Vidakovic. New York, NY: Springer New York, 1999, pp. 291–308. ISBN: 978-1-4612-0567-8 (cit. on p. 23).

[24] Chaofei Yang, Qing Wu, Hai Li, and Yiran Chen. *Generative Poisoning Attack Method Against Neural Networks*. 2017. arXiv: 1703.01340 [cs.CR] (cit. on p. 15).

[25] Raymond A. Yeh, Chen Chen, Teck Yian Lim, Alexander G. Schwing, Mark Hasegawa-Johnson, and Minh N. Do. *Semantic Image Inpainting with Deep Generative Models*. 2017. arXiv: 1607.07539 [cs.CV] (cit. on p. 27).

[26] Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. "Foolbox Native: Fast adversarial attacks to benchmark the robustness of machine learning models in PyTorch, TensorFlow, and JAX." In: *Journal of Open Source Software* 5.53 (2020), p. 2607 (cit. on p. 34).

[27] Lianli Gao, Qilong Zhang, Jingkuan Song, and Heng Tao Shen. *Patch-wise++ Perturbation for Adversarial Targeted Attacks*. 2021. arXiv: 2012.15503 [cs.CV] (cit. on p. 38).

[28] Fei Zuo and Qiang Zeng. *Exploiting the Sensitivity of L_2 Adversarial Examples to Erase-and-Restore*. 2020. arXiv: 2001.00116 [cs.CV] (cit. on pp. 23, 66, 74).

[29] R. Austin McEver and B. S. Manjunath. *PCAMs: Weakly Supervised Semantic Segmentation Using Point Supervision*. 2020. arXiv: 2007.05615 [cs.CV] (cit. on p. 43).

[30] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. "Image Inpainting for Irregular Holes Using Partial Convolutions." In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss. Cham: Springer International Publishing, 2018, pp. 89–105. ISBN: 978-3-030-01252-6 (cit. on pp. 29, 30).

[31] Saining Xie and Zhuowen Tu. *Holistically-Nested Edge Detection*. 2015. arXiv: 1504.06375 [cs.CV] (cit. on p. 44).

[32] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, and Philip H. S. Torr. "Deeply Supervised Salient Object Detection with Short Connections." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.4 (2019), pp. 815–828 (cit. on pp. 41–45, 50, 51, 58, 59).

[33] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. *Generative Face Completion*. 2017. arXiv: 1704.05838 [cs.CV] (cit. on p. 27).

[34] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. "Places: A 10 million Image Database for Scene Recognition." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017) (cit. on pp. 34, 76).

[35] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. "Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks." In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2018) (cit. on p. 33).

[36] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. "Globally and Locally Consistent Image Completion." In: *ACM Trans. Graph.* 36.4 (July 2017). ISSN: 0730-0301 (cit. on p. 27).

[37] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Nets." In: *Advances in Neural Information Processing Systems 27*. Vol. 27. 5. 2014, pp. 2672–2680 (cit. on p. 27).

[38] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. *Towards the Science of Security and Privacy in Machine Learning*. 2016. arXiv: 1611.03814 [cs.CR] (cit. on pp. 14, 15).

[39] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. *Membership Inference Attacks against Machine Learning Models*. 2017. arXiv: 1610.05820 [cs.CR] (cit. on p. 14).

[40] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. "Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures." In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. CCS '15. Denver, Colorado, USA: Association for Computing Machinery, 2015, 1322–1333. ISBN: 9781450338325 (cit. on p. 14).

[41] R. Rangarajan, R. Venkataramanan, and Siddharth Shah. "Image Denoising Using Wavelets." In: 2002 (cit. on p. 23).

[42] SG Chang, B Yu, and M Vetterli. "Adaptive wavelet thresholding for image denoising and compression." In: *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society* 9.9 (2000), 1532—1546. ISSN: 1057-7149 (cit. on p. 23).

[43] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. "Reconstruction by inpainting for visual anomaly detection." In: *Pattern Recognition* 112 (2021), p. 107706. ISSN: 0031-3203 (cit. on pp. 67–70, 76).

[44] Alexander Bagnall, Razvan Bunescu, and Gordon Stewart. *Training Ensembles to Detect Adversarial Examples*. 2017. arXiv: 1712.04006 [cs.LG] (cit. on p. 22).

[45] Zihao Liu, Qi Liu, Tao Liu, Nuo Xu, Xue Lin, Yanzhi Wang, and Wujie Wen. *Feature Distillation: DNN-Oriented JPEG Compression Against Adversarial Examples*. 2019. arXiv: 1803.05787 [cs.CV] (cit. on p. 23).

[46] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, and Jun Zhu. *Improving Adversarial Robustness via Promoting Ensemble Diversity*. 2019. arXiv: 1901.08846 [cs.LG] (cit. on p. 22).

[47] Xiao Wang, Siyue Wang, Pin-Yu Chen, Yanzhi Wang, Brian Kulis, Xue Lin, and Peter Chin. *Protecting Neural Networks with Hierarchical Random Switching: Towards Better Robustness-Accuracy Trade-off for Stochastic Defenses*. 2019. arXiv: 1908.07116 [cs.LG] (cit. on p. 22).

[48] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. *Intriguing properties of neural networks*. 2014. arXiv: 1312.6199 [cs.CV] (cit. on p. 12).

[49] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9 (cit. on p. 18).

[50] Yexin Duan, Xingyu Zhou, Junhua Zou, Junyang Qiu, Jin Zhang, and Zhisong Pan. "Mask-guided noise restriction adversarial attacks for image classification." In: *Computers & Security* 100 (2021), p. 102111. ISSN: 0167-4048 (cit. on p. 63).

[51] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks." In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2574–2582 (cit. on pp. 19, 20, 70).

[52] Wufeng Xue, Lei Zhang, Xuanqin Mou, and Alan C. Bovik. "Gradient Magnitude Similarity Deviation: A Highly Efficient Perceptual Image Quality Index." In: *IEEE Transactions on Image Processing* 23.2 (2014), pp. 684–695.

[53] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. "Image quality assessment: from error visibility to structural similarity." In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612 (cit. on pp. 70, 97).

[54] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramer, Atul Prakash, Tadayoshi Kohno, and Dawn Song. *Physical Adversarial Examples for Object Detectors*. 2018. arXiv: 1807.07769 [cs.CR] (cit. on p. 103).

[55] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. "Robust Physical-World Attacks on Deep Learning Visual Classification." In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1625–1634 (cit. on p. 103).

[56] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4 (cit. on p. 70).

[57] Tie Liu, Zejian Yuan, Jian Sun, Jingdong Wang, Nanning Zheng, Xiaoou Tang, and Heung-Yeung Shum. "Learning to Detect a Salient Object." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.2 (2011), pp. 353–367 (cit. on p. 45).

[58] I. Evtimov, Kevin Eykholt, Earlence Fernandes, T. Kohno, B. Li, Atul Prakash, Amir Rahmati, and D. Song. "Robust Physical-World Attacks on Deep Learning Models." In: *arXiv: Cryptography and Security* (2017) (cit. on p. 4).

[59] Huaizu Jiang, Jingdong Wang, Zejian Yuan, Yang Wu, Nanning Zheng, and Shipeng Li. "Salient Object Detection: A Discriminative Regional Feature Integration Approach." In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2083–2090 (cit. on p. 45).

[60] Oscar Knagg. *Know your enemy: How you can create and defend against adversarial attacks*. 2019. URL: https://towardsdatascience.com/know-your-enemy-7f7c5038bdf3 (visited on 01/06/2019) (cit. on p. 19).

[61] Ilija Mihajlovic. *Everything You Ever Wanted To Know About Computer Vision.* 2020. URL: https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e (visited on 01/06/2020) (cit. on p. 2).

[62] Nicholas Carlini. *Adversarially non-Robust Machine Learning*. 2021. URL: https://www.youtube.com/watch?v=qgsmd2LaZA4 (visited on 01/08/2021) (cit. on pp. 5, 9, 99, 105).

[63] Nicholas Carlini. *A Complete List of All (arXiv) Adversarial Example Papers*. 2021. URL: https://nicholas.carlini.com/writing/2019/all-adversarial-example-papers.html (visited on 01/08/2021) (cit. on pp. 2, 104).

[64] Ge Zheng. *Tensorflow implementation for CVPR paper "Deeply Supervised Salient Object Detection with Short Connections"*. 2021. URL: https://github.com/Joker316701882/Salient-Object-Detection (visited on 08/01/2021) (cit. on pp. 42, 50).

[65] Ian Goodfellow, Nicolas Papernot, Sandy Huang, Rocky Duan, Pieter Abbeel, and Jack Clark. *Attacking Machine Learning with Adversarial Examples*. 2017. URL: https://openai.com/blog/adversarial-example-research/ (visited on 01/01/2017) (cit. on p. 22).

[66] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. "Generative Image Inpainting with Contextual Attention." In: *arXiv preprint arXiv:1801.07892* (2018) (cit. on pp. 23, 27–29, 34, 35, 55, 76, 81).

[67] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. "Free-Form Image Inpainting with Gated Convolution." In: *arXiv preprint arXiv:1806.03589* (2018) (cit. on pp. 29, 30, 35, 55, 76).

[68] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. "ImageNet Large Scale Visual Recognition Challenge." In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252 (cit. on p. 12).

[69] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV] (cit. on p. 13).

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV] (cit. on p. 13).

[71] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. *Rethinking the Inception Architecture for Computer Vision*. 2015. arXiv: 1512.00567 [cs.CV] (cit. on p. 13).

[72] Sébastien Bubeck. *Convex Optimization: Algorithms and Complexity*. 2015. arXiv: 1405.4980 [math.OC] (cit. on p. 19).

[73] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. *Explaining and Harnessing Adversarial Examples*. 2015. arXiv: 1412.6572 [stat.ML] (cit. on pp. 11, 18, 21, 104).

[74] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML] (cit. on p. 22).