DEGREE PROJECT IN ELECTRICAL ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2019

Explainable AI as a Defence Mechanism for Adversarial Examples

HARALD STIFF

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Explainable AI as a Defence Mechanism for Adversarial Examples

HARALD STIFF

Master in Computer Science
Date: June 18, 2019
Supervisor: Joel Brynielsson (KTH), Linus Luotsinen (FOI)
Examiner: Olle Bälter
Swedish title: Förklarbar AI som en försvarsmekanism mot motstridiga exempel
School of Electrical Engineering and Computer Science

Abstract

Deep learning is the gold standard for image classification tasks. With its introduction came many impressive improvements in computer vision outperforming all of the earlier machine learning models. However, in contrast to the success it has been shown that deep neural networks are easily fooled by adversarial examples, data that have been modified slightly to cause the neural networks to make incorrect classifications. This significant disadvantage has caused an increased doubt in neural networks and it has been questioned whether or not they are safe to use in practice. In this thesis we propose a new defence mechanism against adversarial examples that utilizes the explainable AI metrics of neural network predictions to filter out adversarial examples prior to model interference. We evaluate the filters against various attacks and models targeted at the MNIST, Fashion-MNIST, and Cifar10 datasets. The results show that the filters can detect adversarial examples constructed with regular attacks but that they are not robust against adaptive attacks that specifically utilizes the architecture of the defence mechanism.

Sammanfattning

Djupinlärning är den bästa metoden för bildklassificeringsuppgifter. Med dess introduktion kom många imponerande förbättringar inom datorseende som överträffade samtliga tidigare maskininlärningsmodeller. Samtidigt har det i kontrast till alla framgångar visat sig att djupa neuronnät lätt luras av motstridiga exempel, data som har modifierats för att få neurala nätverk att göra felaktiga klassificeringar. Denna nackdel har orsakat ett ökat tvivel gällande huruvida neuronnät är säkra att använda i praktiken. I detta examensarbete föreslås en ny försvarsmekanism mot motstridiga exempel som utnyttjar förklarbar AI för att filtrera bort motstridiga exempel innan de kommer i kontakt med modellerna. Vi utvärderar filtren mot olika attacker och modeller riktade till MNIST-, Fashion-MNIST-, och Cifar10-dataseten. Resultaten visar att filtren kan upptäcka motstridiga exempel konstruerade med vanliga attacker, men att de inte är robusta mot adaptiva attacker som specifikt utnyttjar försvarsmekanismens arkitektur.

Contents

Symbols

F	Neural network
θ	Neural network parameters
y	Output probabilities of neural network
x	Input to neural network
x'	Adversarial example
δ	$x' - x$
$J_\theta(\cdot)$	Loss function of F
$\|\|\cdot\|\|_p$	ℓ_p norm
$\mathbb{1}(\cdot)$	Element-wise indicator function
$\mathbb{E}(\cdot)$	Expected value
$C(\cdot)$	Output class of neural network
$Z(\cdot)$	Logits of neural network
$\mathcal{T}$	Distribution of transformations
$\odot$	Element-wise multiplication
∇	Gradient
$[\cdot]_+$	$\max(0, \cdot)$
$[\cdot]_-$	$\min(0, \cdot)$
I	Identity matrix
e_i	Column basis vector

Chapter 1

Introduction

In recent years, deep neural networks have achieved significant success on image classification tasks. Most notably the usage of deep convolutional neural networks in the ImageNet Large Scale Visual Recognition challenge [21] resulted in drastic improvements in comparison to earlier state-of-the-art image classifiers. Deep neural networks have also been empirically shown to be powerful in a wide range of other tasks such as speech recognition, object detection, and translation services [11, 20, 26]. The reason neural networks work so well can be explained by the massive amount of parallel nonlinear computations used to make a classification. However, this comes with several flaws as the networks can have counterintuitive properties and are difficult to interpret. Even though there is a reason to believe that neural networks generalize well there is one counterintuitive property in particular that reduces the confidence in neural networks. It has been shown that neural network image classifiers are easily fooled by *adversarial examples* [27], data that have been slightly transformed to cause neural networks to misclassify them while still being visually indistinguishable from the original data. Adversarial examples can be constructed in many different ways, two of which are shown in Figure 1.

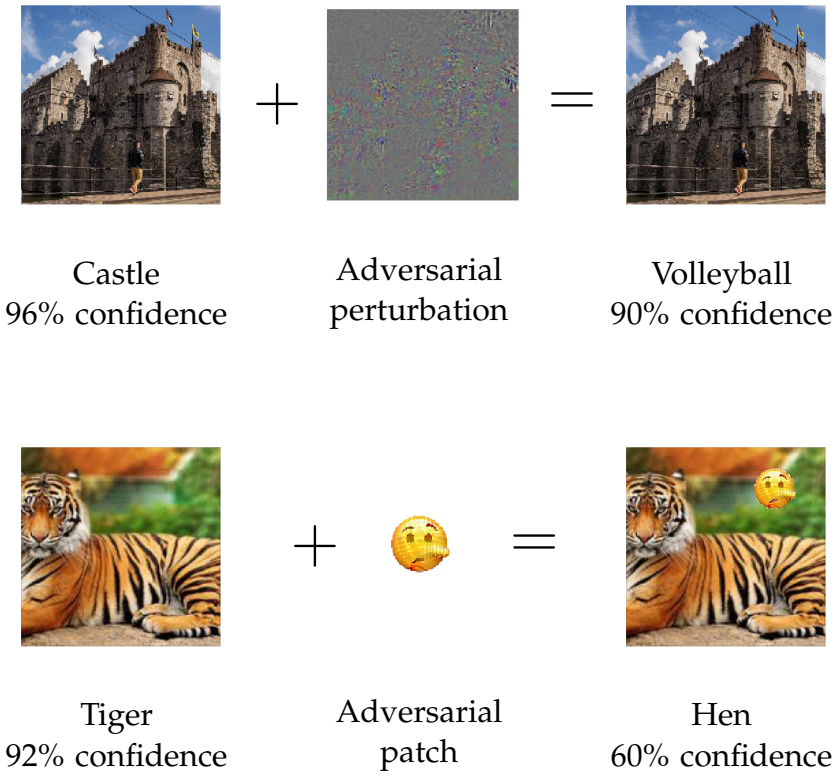


Figure 1.1: Output predictions of the VGG16 neural network [25] for original images (left) and images corrupted with adversarial attacks (right).

The degree to which an attacker can fool a neural network vastly limits the domains in which the network can be used. The discovery of adversarial examples has inspired researchers to develop defence mechanisms that robustify neural networks to the attacks. Although many defences have been proposed [15, 29, 33] with methods ranging from hiding the model parameters to randomizing the neural network the threat of adversarial examples still remains. The defences are often easily bypassed with adaptive attacks that make use of the defence strategies [2]. Recent work [23] have even provided reasons to believe that adversarial examples are inevitable with probabilistic reasoning. The reason why adversarial examples exist is an open research topic, but many researchers tend to agree that the large number of dimensions of the input eases the creation of adversarial examples [9, 23].

1.1 Research Question

Explainable AI has gained a lot of attention recently as a possible way to get better insight into how neural networks classifies data. Explainable AI are models whose decisions are easily understood by humans [10]. In contrast to black-box models, explainable AI seeks to provide explanations to predictions. For image classifiers, this could mean to highlight segments of an image that caused the prediction to be made. Little work has been conducted on whether or not explainable AI techniques can be used to filter out adversarial examples. Wu et al. [31] has shown that the explainable AI metrics are altered when an adversarial perturbation is added to an image. It is intuitive to believe that adversarial examples fool neural networks by shifting the attention regions within an image that are captured with explainable AI models. This thesis aims to draw the connection between the shifted attention regions and adversarial examples by answering the following research question:

- Can explainable AI techniques be used to identify patterns in adversarially crafted image data?

The interest in finding a robust defence to adversarial examples is massive. Adversarial attacks pose a critical threat that must be addressed for neural networks used in situations where safety is a major concern. As the world becomes more autonomous with large machines using neural networks for decision making it is of great importance that the networks cannot be fooled.

Chapter 2

Background

This project has been done in collaboration with the *Swedish Defence Research Agency* (FOI). Their vision is to research for a safer and more secure world by providing cutting-edge research and expertise in defence and security. As there is rapid development and interest in Data Science and AI it is inevitable that FOI wants to be a part of state-of-the-art technologies within these fields. With rapid technological advances, there is always a risk for unwanted exploitation of the techniques that can cause great harm for the world which this project will highlight. FOI's interest in this work is to gain knowledge of the flaws of deep neural networks, how they can be fooled and to what extent damage can be caused by an attacker. Since deep neural networks are becoming increasingly present in many systems where safety is critical there is a need for guidelines and safety measures that ought to be enforced in order to prevent attackers from causing harm.

Chapter 3

Theory

The theory section is split into 5 main parts. An introduction to the notation used throughout the report and an introduction to adversarial examples are presented in Sections 3.1 and 3.2. Section 3.3 provides definitions and metrics of robustness. The most prominent adversarial attacks and defences are presented in Sections 3.4 and 3.5. Lastly, explainable AI techniques are described in Section 3.6.

3.1 Neural Network Notation

A neural network is a function $F(x, \theta) = y$ where $x \in \mathbb{R}^n$ is the input vector, $y \in \mathbb{R}^m$ is the output vector and $\theta = \{(W_i, b_i)\}_{i=1}^{l}$ are the parameters of the network [35]. We consider neural networks with the property that $\sum_{i=1}^{m} y_i = 1$ and $0 \leq y_i \leq 1$ meaning that the input is mapped to a probability distribution. The mapping is done through a number of layers

$$F(x, \theta) = \text{softmax} \circ F_l \circ F_{l-1} \circ \ldots \circ F_1(x) \tag{3.1}$$

where $F_i(x) = \sigma(W_i x + b_i)$ and $\sigma(\cdot)$ is a nonlinear activation function (ReLU, sigmoid, tanh etc.) [6]. The softmax function maps its input to a probability distibution with

$$\text{softmax}(z)_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \tag{3.2}$$

where the temperature T usually is set to 1. The neural network is a classifier, it assigns a label to each input x with a function $C(x)$ where

$$C(x) = \underset{i}{\text{argmax}}\, F(x, \theta)_i. \tag{3.3}$$

Each input has a correct label $C^*(x)$. The parameters θ of F are trained to make $C(x)$ as close as possible to $C^*(x)$ for all inputs x [35]. This is ensured by finding an approximate solution to

$$\underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{(x,l)\sim\mathcal{D}}[J_\theta(x,l)] \tag{3.4}$$

where J is a loss function that measures how well the F classifies inputs and $\mathcal{D} = \{(x_i, l_i)\}_{i=1}^N$ is a dataset of inputs x_i and their correct labels l_i. Moreover we define the input to the softmax function as $Z(x)$ where $F(x) = \text{softmax}(Z(x))$. These are also called the logits of the neural network.

In many cases the input x to the neural network is an image. In this report we will represent images as vectors $x \in [0,1]^d$ where $d = h \times w \times c$ (height $\times$ width $\times$ number of channels) [28].

3.2 Adversarial Examples

Given an input x with a label $C(x) = C^*(x) = l$, an adversarial example is an input $x' = x + \delta$ that is close to x under some distance metric $d(x, x')$ where $C(x') \neq l$ [7]. Adversarial examples are constructed to be misclassified by the neural network while still being visually indistinguishable from the original input x. The attacks are separated in many different classes. First and foremost the attacks can be targeted or non-targeted. In a targeted attack, the objective is to create an adversarial example x' that gets classified as a specifically chosen label l' where $l' \neq C^*(x)$. In contrast to this, nontargeted attacks only aim to make the adversarial example misclassified with no further restriction on what the misclassified label should be. The targeted attack is in general a harder problem to solve than the nontargeted attack as the adversarial examples must satisfy more constraints. Adversarial attacks are also categorized into white box and black box attacks. In a white box attack, all of the neural network parameters are known to the attacker whereas only the output probabilities y are known to the attacker in a black box attack.

As a measure of proximity between x and x' the ℓ_p norm is used where

$$||x - x'||_p = \left(\sum_{i=1}^{n} |x_i - x_i'|^p \right)^{\frac{1}{p}}. \tag{3.5}$$

When $p = \infty$, $||x - x'||_\infty = \max_i |x_i - x'_i|$ which is the largest difference between two elements of x and x'. Morover, when $p = 0$, $||x - x'||_0 = \sum_{i=1}^{n} \mathbb{1}(x_i \neq x'_i)$ meaning that the ℓ_0 norm yields the number of elements that differs between x and x'. Adversarial examples are constructed to minimize the ℓ_p distance between x and x' for different values of p while having x' classified incorrectly by the neural network. In general, adversarial examples are constructed by solving the optimization problem

$$\begin{aligned} \text{minimize } & ||\delta||_p \\ \text{s.t } \quad & C(x) = l \\ & C(x + \delta) = l' \\ & l \neq l' \\ & x + \delta \in [0, 1]^n. \end{aligned} \tag{3.6}$$

However, the constraints in Eq. (3.6) are highly nonlinear making it a difficult problem to solve. In practice, adversarial examples are constructed in many different ways, the most common attack methods will be presented in Section 3.4.

3.3 Robustness

In order to defend neural networks against adversarial attacks, it is essential to precisely define what attacks the networks must be resilient to. This section will use the definitions by Madry et al. [18] to do this. Also, fundamental conditions on when neural networks cannot be robust will be provided.

3.3.1 Adversarial Robustness

To ensure robustness to adversarial examples with perturbations belonging to a set $\mathcal{S}$ we aim to train a neural network $F(x, \theta)$'s parameters such that the saddle point problem

$$\underset{\theta}{\operatorname{argmin}} \, \mathbb{E}_{(x,l)\sim\mathcal{D}} \left[\max_{\delta \in \mathcal{S}} \left(J_\theta(x + \delta, l) \right) \right] \tag{3.7}$$

is solved. This objective ensures to minimize the expected maximum increase in loss that can be caused by an adversarial example in a set

$\mathcal{S}$. The objective does not only give a clear objective of what a robust classifier should satisfy but also gives a quantitative measure of how robust the network is for data $(x, l) \in \mathcal{D}$. Moreover an input x is defined to be *epsilon locally robust* to a given norm p if and only if

$$||\delta||_p \leq \varepsilon \implies C(x) = C(x + \delta) \ \forall \varepsilon. \tag{3.8}$$

This robustness measure ensures that an input x will be classified the same for all ℓ_p-bounded perturbations δ.

3.3.2 Bounds on Susceptibility of a Classifier to Adversarial Examples

As no defence mechanisms today are fully robust to adversarial attacks it is natural to raise the question of whether or not adversarial examples are inevitable. This is discussed by Shafahi et al. [23] where theoretical bounds on the susceptibility of classifiers to adversarial examples are provided. In the paper the following theorem was derived:

Theorem 1 *Consider a classification problem with m classes, each distributed over the unit hypercube $[0, 1]^n$ with density functions $\{\rho_c\}_{c=1}^m$. Choose a classifier function $C : [0, 1]^n \to \{1, 2, ..., m\}$ that partitions the hypercube into disjoint measurable subsets. Define the following scalar constants:*

- *Let U_c denote the supremum of ρ_c*
- *Let f_c be the fraction of the hyper cube partitioned into class c by C.*

Choose some class c with $f_c \leq \frac{1}{2}$, and select an ℓ_p-norm with $p \geq 2$. Sample a random data point x from the class distribution ρ_c. Then with probability at least

$$1 - U_c \frac{\exp(-\pi\varepsilon^2)}{2\pi} \tag{3.9}$$

one of the following conditions hold:

- *x is misclassified by C or,*
- *x has an adversarial example x', with $||x - x'||_p \leq \varepsilon$.*

The theorem explicitly limits the possibility to ensure robustness for classification tasks where the input data distribution has low values of U_c. For small values of U_c the probability in Eq. (3.9) will approach 1 even for small values of ε.

3.4 Adversarial Attacks

This section will provide the theory for a wide range of attacks methods. It will be shown that neural networks are vulnerable to a wide range of attacks. Not only are the attacks different in how they construct adversarial examples but they also differ in attack objectives. In the following sections both the theory of the attacks and the objective the attacks are meant to fulfill will be described.

3.4.1 Fast Gradient Sign Method

The fast gradient sign method (FGSM) [9] is the earliest known attack to be presented. It constructs adversarial examples with ℓ_∞-norms bounded by ε with

$$x' = x + \varepsilon \text{sign}(\nabla J_\theta(x, l)). \tag{3.10}$$

The attack was discovered by considering the dot product between a weight vector w and an adversarial example $x' = x + \delta$:

$$w^T x' = w^T x + w^T \delta. \tag{3.11}$$

If δ is chosen such that $\delta = \text{sign}(w)$ the distortion $w^T\delta$ grows linearly with the dimension of x. By approximating the loss function $J_\theta(x, l)$ as a linear function one can find the direction of the distortion that maximally increases the linearized loss as $\delta = \varepsilon \text{sign}(\nabla_x J_\theta(x, l))$. This method does not produce minimal adversarial perturbations but rather produces adversarial examples quickly as this method only requires a gradient to be computed once.

3.4.2 Projected Gradient Descent

The projected gradient descent attack (PGD) is an extension of the FGSM attack. Instead of taking one step in the direction of the gradient this attack takes multiple smaller steps with

$$x'_0 = x \tag{3.12}$$

$$x'_{t+1} = \text{clip}_{[0,1]}\left[x'_t + \alpha \text{sign}(\nabla J_\theta(x'_t, l))\right] \tag{3.13}$$

where $\text{clip}_{[0,1]}(\cdot)$ is a function that forces its input to be in the range of $[0, 1]$. By recomputing the gradient at each step the attack is much more computationally expensive but also a stronger attack than the FGSM as it moves the input more precisely along the gradient.

3.4.3 Carlini & Wagner's Attack

This attack was constructed by slightly altering the optimization problem in Eq. (3.6) [6]. Instead of imposing a hard constraint on x' to be classified as l' a soft constraint is imposed instead with the help of a Lagrangian multiplier. The new optimization problem is

$$\begin{aligned} &\underset{\delta}{\text{argmin}} ||\delta||_p + cf(x+\delta) \\ &\text{s.t } x+\delta \in [0,1]^n \end{aligned}$$

where f is an objective function such that if $C(x+\delta) = l'$ if and only if $f(x+\delta) \leq 0$ and c is constant that is chosen iteratively. It is shown empirically that the optimal constant c is chosen as the smallest value that yields a solution δ^* such that $f(x+\delta^*) \leq 0$. The box constraint $x+\delta \in [0,1]^n$ can be eliminated with a change of variable $\delta = \frac{1}{2}(\tanh(w)+1) - x$ giving a new optimization objective

$$\underset{w}{\text{argmin}} ||\frac{1}{2}(\tanh(w)+1) - x||_p + cf\left(\frac{1}{2}(\tanh(w)+1)\right). \tag{3.14}$$

This is an iterative attack solved with the gradient descent algorithm. It is much slower than Eq. (3.10) as it requires a gradient for each iteration of the algorithm but constructs smaller adversarial perturbations.

3.4.4 Expectation Over Transformation

In order for adversarial examples to be a real threat to deep learning systems they must be robust to natural transformations such as camera noise, changes in lightning, rotations, etc. This attack which was developed by Athalye et al. [3] constructs adversarial examples that are robust over en entire distribution of transformations that are present in the real physical world. The attack constructs adversarial examples by finding a solution to

$$\begin{aligned} &\underset{x'}{\text{argmax}}\, \mathbb{E}_{t\sim\mathcal{T}}\Big[\log(y_t|t(x'))\Big] \\ &\text{s.t } \mathbb{E}_{t\sim\mathcal{T}}||t(x)-t(x')||_p < \varepsilon \\ &x' \in [0,1]^d \end{aligned} \tag{3.15}$$

where $\mathcal{T}$ is a distribution of transformations. Instead of finding one instance that maximizes the objective this method finds an adversarial example that stays adversarial to the transformations defined in $\mathcal{T}$. An approximate solution to Eq. (3.15) is obtained by using a Lagrangian multiplier as in Carlini and Wagner's method giving the new objective

$$\underset{x'}{\text{argmax}}\ \mathbb{E}_{t\sim\mathcal{T}}\Big[\log(y_t|t(x')) - \lambda||t(x) - t(x')||_p\Big] \tag{3.16}$$

that can be solved with the gradient descent algorithm. An example of the attack is shown in Figure 3.1.

Figure 3.1: The images furthest to the left show two adversarial examples, the top one constructed with the expectation over transformation (EOT) attack and the bottom one with the projected gradient descent (PDG) attack. To the right of both of the adversarial examples are four different transformations of the images. A checkmark indicates a successful attack and a cross indicates a failed attack.

This attack can also be extended to create adversarial patches. Instead of manipulating the image itself the attack can be used to create a

smaller patch that will cause the image to be misclassified. One example usage of this is creating stickers that will make deep learning systems missclassify road signs. Let $p' \in [0,1]^n$ where $||p'||_0 = d < n$ be the adversarial patch, let $p \in [0,1]^n$ be the image it is aimed to look similar to and let x be the image that is to be manipulated by the patch. The adversarial example is constructed by placing the patch on the image

$$x' = x \odot (1 - \mathbb{1}(p' > 0)) + p' \tag{3.17}$$

where $\odot$ represents an element-wise multiplication. The patch p' is optimized to maximize the target output classification of x' while still keeping p' looking visually close to p. The full objective is

$$\begin{aligned} \underset{p'}{\operatorname{argmax}} \; & \mathbb{E}_{x\sim\mathcal{D}}\, \mathbb{E}_{t\sim\mathcal{T}} \left[\log \left(y_t \middle| x \odot (1 - \mathbb{1}(t(p') > 0)) + t(p') \right) \right] \\ \text{s.t} \quad & \mathbb{E}_{t\sim\mathcal{T}}\, ||t(p') - t(p)||_2 \leq \varepsilon \\ & p' \in [0,1]^n. \end{aligned} \tag{3.18}$$

This attack is in principle the same as Eq. (3.15) where p' is trained to be robust over the transformations defined in $\mathcal{T}$. In addition p' is trained to robustly misclassify multiple images in a dataset $\mathcal{D}$. In Figure 3.2 six different adversarial patches and their target labels are shown. The stickers were optimized to make images adversarial to the VGG16, VGG19, and the Xception neural networks when the stickers are applied to the images. Similar attacks are shown in Figure 3.3 but where the stickers are disguised as emojis.

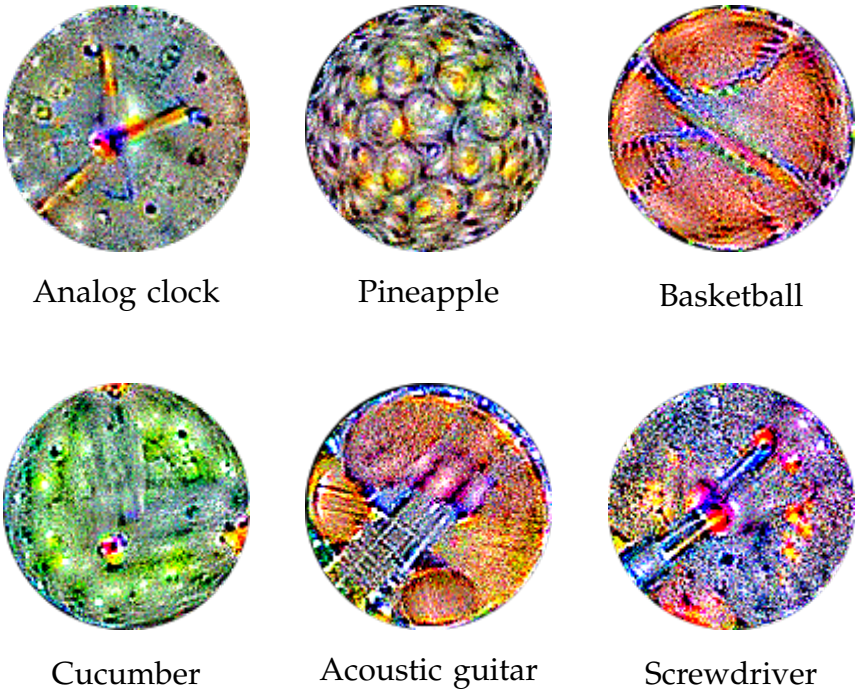


Figure 3.2: Adversarial stickers and their target labels.

Figure 3.3: Adversarial stickers disguised as emojis and their target labels.

3.4.5 Black Box Attacks

It is rare to find systems using neural networks where the model parameters are fully accessible to anyone wanting them. In order for the attacks to be conducted in practice they must work in a black box setting meaning that only the model output probabilities $y \in \mathbb{R}^m$ are

accessible to the attacker. Black box attacks work because of the transferable property of adversarial examples. Given two different neural networks $F^1(x, \theta_1)$ and $F^2(x, \theta_2)$ trained to learn the same classes it is empirically shown that adversarial examples generated with the parameters θ_1 of F^1 in many cases remain adversarial to the second network F^2 [17]. This is especially prominent in non-targeted adversarial examples. The same transferability can be enhanced with the approach by Liu Liu et al. [17] by generating adversarial examples that are largely model agnostic. Such adversarial examples can be generated with

$$\underset{x'}{\operatorname{argmin}} - \log\left(\sum_{i=1}^{N} \alpha_i J_{\theta_i}(x', l')\right) + \lambda||x - x'||_p \tag{3.19}$$

where x' is optimized to be misclassified by N different neural networks. In the case of a black-box attack it is therefore often enough to construct adversarial examples by using any other network that has been trained to learn the same classes. If such networks are not available one can be constructed by training a distilled network. A distilled network $F^{\text{dist}}(x, \theta_d)$ is trained to output the same output probabilities as the neural network F^{target} it is aimed to replicate, instead of being trained on hard coded labels. By constructing a dataset $\mathcal{D} = \{(x_i, F^{\text{target}}(x_i, \theta_t)\}_{i=1}^{N}$ where the the softmax function Eq. (3.3) of F^{target} is altered by using a higher temperature T the distilled network's parameters θ_d are obtained with Eq. (3.4).

3.5 Defence Mechanisms

As it is clear that neural networks are vulnerable to many different forms of adversarial attacks it is natural to ask what methods of defence exists to tackle the attacks. This section will provide a description of some of the proposed defence mechanisms in the literature.

3.5.1 Adversarial Retraining

This approach developed by Goodfellow, Shlens, and Szegedy [9] was the first defence developed to tackle adversarial examples. It works simply by adding adversarial inputs and their correct labels to the

training data when training the neural network. In contrast to the original objective in Eq. (3.4) a new weighted objective

$$\underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{(x,l)\sim\mathcal{D}}\, \alpha J_\theta(x,l) + \mathbb{E}_{(x',l')\sim\mathcal{D}'}\, (1-\alpha) J_\theta(x',l') \tag{3.20}$$

is used where $\mathcal{D}'$ is a dataset consisting of adversarially crafted inputs and their corresponding labels. The constant α is used to determine how much emphasis should be placed on correctly classifying the adversarial inputs.

3.5.2 Obfuscated Gradients

Close to all of the attacks used against neural networks require a gradient to be computed. This defence method aims to mask the gradient in such a way that it becomes unusable when constructing an attack. This gradient masking can either cause the gradient to be stochastic or numerically unstable. Consider a trained neural network $F(x,\theta)$. The defence works by creating a new classifier $\hat{F}(x,\theta) = F(g(x),\theta)$ where $g(x) \approx x$. The function g is constructed to be close to the identify function while being neither smooth or differentiable thus keeping the functionality of the original classifier but hiding the gradients.

3.5.3 Convex Outer Adversarial Polytope

This is a defence that provably reduces the adversarial loss

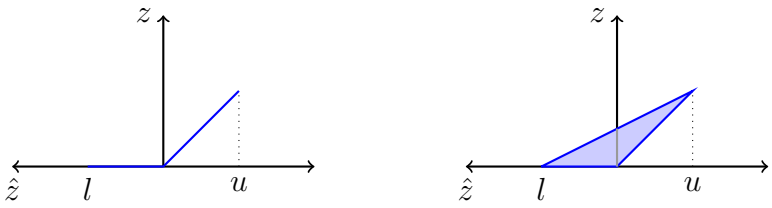
$$\mathbb{E}_{(x,y)\sim\mathcal{D}} \max_{||\delta||_\infty \leq \varepsilon} J_\theta(x+\delta, l) \tag{3.21}$$

for ReLU-based neural networks [13]. It works by considering the set of logits reachable from a set of perturbed inputs and approximating it with a convex set. Let

$$\mathcal{Z}_\varepsilon(x) = \{Z(x+\delta) : ||\delta||_\infty \leq \varepsilon\} \tag{3.22}$$

be the adversarial polytope, the set of reachable logits by perturbing the input x with δ that has an ℓ_∞ norm bounded by ε. As this is a nonconvex set, a new set $\tilde{\mathcal{Z}}_\varepsilon(x)$, the convex outer bound of $\mathcal{Z}_\varepsilon(x)$ is considered. $\tilde{\mathcal{Z}}_\varepsilon(x)$ is constructed with linear relaxations of the ReLU-function as illustrated in Figure 3.4. Given lower and upper bounds l, u of the pre-ReLU activations the convex relaxation set is given by

$$z \geq 0,\ z \geq \hat{z},\ -u\hat{z} + (u-l)z \leq -ul. \tag{3.23}$$



ReLU set Convex relaxation

Figure 3.4: An illustration of the convex relaxation of the ReLU-function given an upper and lower bound u, l.

Using $\tilde{\mathcal{Z}}_\varepsilon(x)$ it is possible to verify the robustness of a neural network by finding a solution to the linear program

$$\begin{aligned} \min \ & Z_l - Z_{l'} = c^T Z \\ \text{s.t } & Z \in \tilde{\mathcal{Z}}_\varepsilon(x) \end{aligned} \tag{3.24}$$

where Z_l is the logit corresponding to the correct classification and $Z_{l'}$ is the logit corresponding to some other class. If the objective is larger than zero for all classes $l' \neq l$ the neural network is epsilon locally robust with respect to the input x. However, since Eq. (3.24) is not possible to solve in practice the dual problem is considered instead:

$$\begin{aligned} & \max_{\alpha} \mathcal{J}_\varepsilon(x, G_\theta(c, a)) \\ & \text{s.t } \alpha_{i,j} \in [0, 1] \ \ \forall i, j \end{aligned} \tag{3.25}$$

where

$$\mathcal{J}_\varepsilon(x, \nu) = -\sum_{i=1}^{k-1} \nu_{i+1}^T b_i - x^T \hat{\nu}_1 - \varepsilon ||\hat{\nu}_1||_1 + \sum_{i=2}^{k-1} \sum_{j \in \mathcal{I}_i} l_{i,j} [\nu_{i,j}]_+ \tag{3.26}$$

and $G_\theta(c, a)$ is a k-layered neural network defined by

$$\begin{aligned} \nu_k &= -c \\ \hat{\nu}_i &= W_i^T \nu_{i+1}, \quad i = k-1, ..., 1 \\ \nu_{i,j} &= \begin{cases} 0 & \text{for } j \in \mathcal{I}_i^- \\ \hat{\nu}_{i,j} & \text{for } j \in \mathcal{I}_i^+ \\ \frac{u_{i,j}}{u_{i,j} - l_{i,j}} [\hat{\nu}_{i,j}]_+ - \alpha_{i,j} [\hat{\nu}_{i,j}]_- & \text{for } j \in \mathcal{I}_i. \end{cases} \end{aligned} \tag{3.27}$$

$\mathcal{I}_i^+, \mathcal{I}_i^-, \mathcal{I}_i$ denotes the set of activations in layer i where the upper and lower bounds are both positive, negative and span zero respectively. Any feasible solution to the dual problem in Eq. (3.25) provides a lower bound to the primal problem in Eq. (3.24). By setting $\alpha_{i,j} = \frac{u_{i,j}}{u_{i,j} - l_{i,j}}$ the entire backwards pass becomes a linear function. For increased efficiency Eq. (3.25) can be computed for $c = I$ and $c = -I$ to obtain upper and lower bounds for all coefficients. For $c = I$ the backwards pass in Eq. (3.27) becomes

$$\begin{aligned} \hat{\nu}_i &= -W_i^T D_{i+1} W_{i+1}^T \dots D_n W_n^T \\ \nu_i &= D_i \hat{\nu}_i \end{aligned} \tag{3.28}$$

where

$$(D_i)_{jj} = \begin{cases} 0 & j \in \mathcal{I}_i^- \\ 1 & j \in \mathcal{I}_i^+ \\ \frac{u_{i,j}}{u_{i,j} - l_{i,j}} & j \in \mathcal{I}_i. \end{cases} \tag{3.29}$$

Before the dual objective can be computed the upper and lower bounds u, l must be provided with algorithm 1 which computes the bounds one layer at a time.

Algorithm 1 Computing Activation Bounds

Require: Network parameters $\{W_i, b_i\}_{i=1}^k$, input x, ball size ε
  // initialization
  $\hat{\nu}_1 \leftarrow W_1^T$
  $\gamma_1 \leftarrow b_1^T$
  $l_2 \leftarrow x^T W_1^T + b_1^T - \varepsilon ||W_1^T||_{1,:}$
  $u_2 \leftarrow x^T W_1^T + b_1^T + \varepsilon ||W_1^T||_{1,:}$
  // $||\cdot||_{1,:}$ for a matrix here denotes ℓ_1 norm of all columns
  for $i = 2, ..., k-1$ **do**
    form $\mathcal{I}_i^-, \mathcal{I}_i^+, \mathcal{I}_i$; form D_i as in Eq. (3.29)
    // initialize new terms
    $\nu_{j,\mathcal{I}_i} \leftarrow (D_i)_{\mathcal{I}_i} W_i^T$
    $\gamma_i \leftarrow b_i^T$
    // propagate existing terms
    $\nu_{j,\mathcal{I}_j} \leftarrow \nu_{j,\mathcal{I}_j} D_i W_i^T$, $j = 2, ..., i-1$
    $\gamma_j \leftarrow \gamma_j D_i W_i^T$, $j = 1, ..., i-1$
    $\hat{\nu}_1 \leftarrow \hat{\nu}_1 D_i W_i^T$
    // compute bounds
    $\psi_i \leftarrow x^T \hat{\nu}_1 + \sum_{j=1}^i \gamma_j$
    $l_{i+1} \leftarrow \psi_i - \varepsilon ||\hat{\nu}_1||_{1,:} + \sum_{j=2}^i \sum_{i' \in \mathcal{I}_i} l_{j,i'} [-\nu_{j,i'}]_+$
    $u_{i+1} \leftarrow \psi_i + \varepsilon ||\hat{\nu}_1||_{1,:} + \sum_{j=2}^i \sum_{i' \in \mathcal{I}_i} l_{j,i'} [\nu_{j,i'}]_+$
  end for
  return bounds $\{l_i, u_i\}_{i=2}^k$

The dual network in Eq. (3.27) can also be used to construct an adversarial loss summarized in the following theorem:

Theorem 2 *Let L be the cross entropy loss function. For any data point (x,l), and $\varepsilon > 0$, the worst case adversarial loss can be upper bounded by*

$$\max_{||\delta||_\infty \leq \varepsilon} L(Z_\theta(x+\delta), l) \leq L(-\mathcal{J}_\varepsilon(x, G_\theta(e_l 1^T - I)), l) \tag{3.30}$$

where $\mathcal{J}_\epsilon$ is vector valued and as defined in Eq. (3.26) *for a given ε and G_θ is as defined in Eq.* (3.27) *for the given model parameters θ.*

Since G_θ is completely defined by Eq. (3.27) a robust loss function can be created from Eq. (3.30) that can be used to guarantee a reduction in the adversarial loss in Eq. (3.21) when used in training where

$$J_{\text{robust}} = \mathbb{E}_{(x,l)\sim\mathcal{D}} L(-\mathcal{J}_\varepsilon(x, G_\theta(e_l 1^T - I)), l). \tag{3.31}$$

3.6 Explainable AI

When designing machine learning models there is a trade-off between model complexity and model interpretability. In many cases, the interpretability suffers with increasing model complexity. Explainable AI are models whose decisions can be understood by humans. These techniques are used to not only understand the predictions made by neural networks but are also used to verify that the neural networks work in the intended way. This section will summarize the most prominent explainable AI models in the literature.

3.6.1 Class Saliency Extraction

One simple way to obtain an explanation to a neural network image classifier is to use the image gradients [24]. By utilizing the fact that a scalar function maximally increases in the direction of the gradient one can find what pixels in an image has the largest impact on the target logit. Let $x_0 \in \mathbb{R}^{n\times m\times c}$ be an image and $Z_l(x)$ be the logit corresponding to the target class. The class saliency map $M \in \mathbb{R}^{n\times m}$ is defined as

$$M_{ij} = \max_c \mathbf{abs}(\nabla_x Z_l(x)|_{x=x_0})_{ijc} \tag{3.32}$$

and can be used as a heat-map to visualize pixels of an image with large image gradient magnitudes.

3.6.2 Grad-CAM

Given any convolutional neural network Grad-CAM provides a gradient-based explanation for any image classification. It is a generalization of the class activation mapping method (CAM) [34] which was restricted to a few convolutional neural network architectures. This method developed by Selvaraju et al. [22] provides a localization map $L^c_{\text{Grad-CAM}}(x) \in \mathbb{R}^{n\times m}$ that spatially explains what parts of an image x were responsible for a given classification. Let

$$\alpha_c^k = \frac{1}{N}\sum_i\sum_j \frac{\partial Z_c(x)}{\partial A^k_{ij}} \tag{3.33}$$

where A^k is the k:th channel of the last convolutional layer. The scalar weights α_c^k represents the importance of each input channel. Using

Eq. (3.33) the Grad-CAM explanations are computed with a weighted sum as

$$L^c_{\text{Grad-CAM}}(x) = \text{ReLU}\left(\sum_k \alpha^c_k A^k\right). \tag{3.34}$$

The result is a heat-map with the same size as the channels of the last convolutional layer. By resizing the heat-map to the size of the input image it can be used to visualize the regions of the image that caused the classification to be made.

Five examples of Grad-CAM heatmaps are shown in Figure 3.5.

Figure 3.5: Images of animals and their Grad-CAM heatmaps. The heatmaps were extracted from the VGG16 neural network.

3.6.3 Layer-Wise Relevance Propagation

The layer-wise relevance propagation (LRP) technique assigns relevance measures R^l_i to each activation x_i of each layer l of the neural network such that

$$F_\theta(x) \approx \sum_{d=1}^{V_L} R^L_d = \sum_{d=1}^{V_{L-1}} R^{L-1}_d = \cdots = \sum_{d=1}^{V_1} R^1_d \tag{3.35}$$

where V_l is the number of activations at layer l [4]. The goal is to find the relevance scores of the first layer R^1_i which shows how each element of the input contributed to the actual output $F_\theta(x)$. Let $R^L = F_\theta(x)$, to obtain the relevance scores of the input the relevance scores

are backpropagated with

$$R_i^l = \sum_j \frac{x_i^l w_{ij}^+}{\sum_i x_i^l w_{ij}^+} R_j^{l+1}. \tag{3.36}$$

for $l = L-1, L-2, \ldots\ 1$ where w_{ij} are the weights of the neural network as defined in 3.1.

Five examples of LRP heatmaps are shown in Figure 3.6.

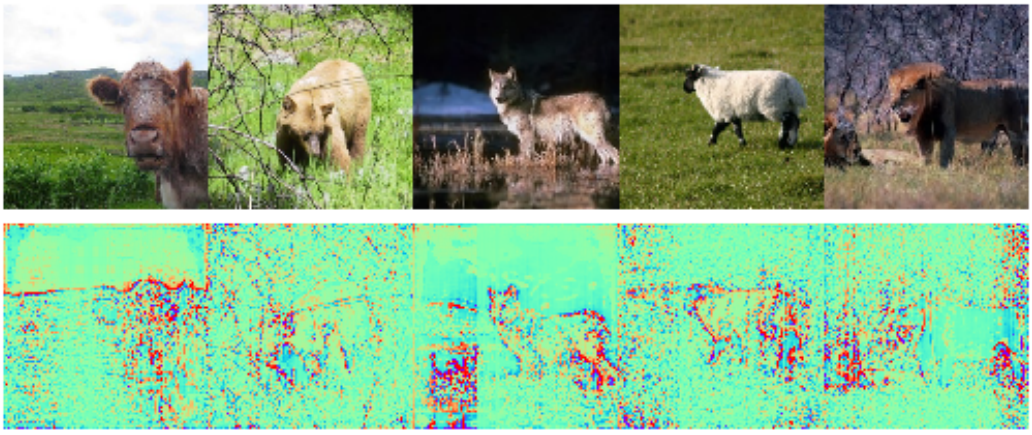


Figure 3.6: Images of animals and their LRP heatmaps. The heatmaps were extracted from the VGG16 neural network.

Chapter 4

Method

This chapter will describe the details of the experiments conducted to test the main hypothesis, that explainable AI can be used to identify patterns in adversarial examples. To evaluate the hypothesis, binary filter neural networks were trained to detect whether or not an image is an adversarial example with the help of its explainable AI metrics. The filters were trained with images constructed using various attacks and evaluated on images constructed with the same attack methods but also images created with attack methods not used in the training set. Furthermore, the filters were trained to detect adversarial examples for several different models and datasets of varying complexity to analyze how well the method scales and how sensitive it is to certain classification tasks.

The intuition behind the approach is discussed in Section 4.1 and the overall architecture of the filters is described in further detail in Section 4.2. The datasets used in the experiments are described in Section 4.3 and the metrics used to measure the performance of the filters is described in Section 4.7.

4.1 Intuition

Since adversarial examples are optimized to remain as similar as possible to the original inputs, there are reasons to believe that features from the original and adversarial inputs remain similar too. However, since their output classifications vastly differ, it is intuitive to believe that the regions of the images that caused the classifications to be made

are changed. This is illustrated with an example in Figure 4.1.

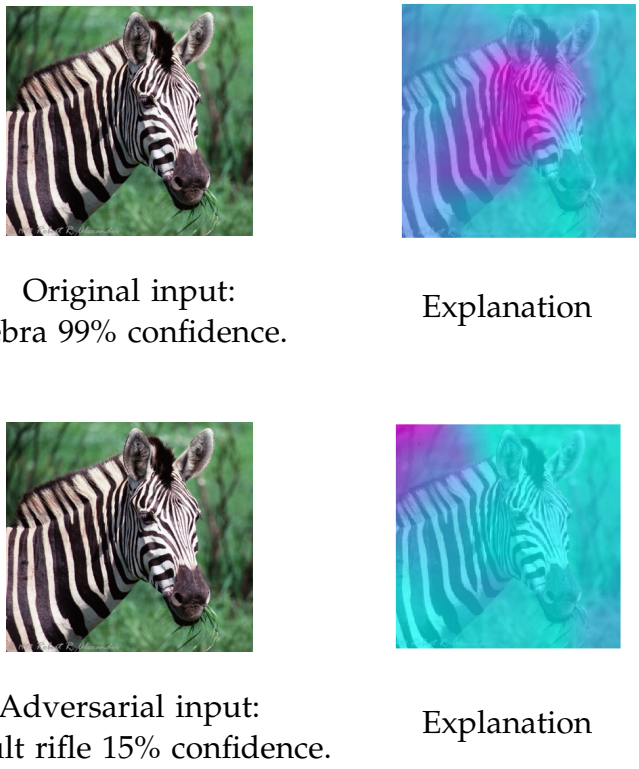


Figure 4.1: Output predictions of the VGG16 neural network [25] for original and adversarial inputs (left) and their corresponding explanations (right). The adversarial input was constructed with Carlini & Wagner's method with an ℓ_∞-distance of 0.05 from the original image. The Grad-CAM method was used to extract the explanations for the classifications.

Even though the inputs are seemingly identical, the explanation of the classification for the adversarial input is shifted. In the original input, the pixels that caused classification to be made were the pixels of the zebra which aligns with the human intuition. However, for the adversarial input, the pixels of interest for the classifier were not the pixels of the zebra but instead the pixels of the background which resulted in an incorrect classification. The objective of the experiments was to test to what extent the heatmaps can be used to detect adversarial examples.

4.2 Adversarial Filter Architecture

The adversarial filter is an extension of a neural network image classifier. It extracts explainable AI heatmaps from the original classifier and uses them as inputs to a second binary network which filters out heatmaps that were extracted from adversarially crafted inputs from heatmaps crafted from original inputs. The general architecture of the adversarial filter model is shown in Figure 4.2.

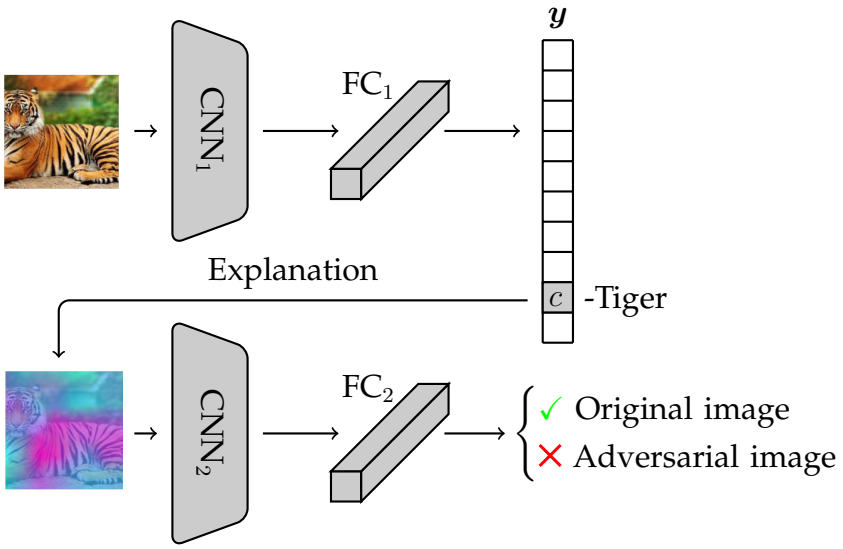


Figure 4.2: Workflow of the adversarial filter. (CNN_1, FC_1) is the image classifier, given its classification of an image an explainable AI method is used to extract an explanation. The explanation is used as an input to the binary filter neural network (CNN_2, FC_2) which is trained to detect if the explanation heatmap was constructed from an adversarial example or not.

The binary networks were trained with two different input strategies. Firstly, as a benchmark, they were trained without heatmaps. The original images and their adversarial examples were used directly as inputs and the networks were trained to directly filter out the adversarial examples without the help of the heatmaps. Secondly, the binary networks were trained using only the heatmaps extracted with various explainable AI techniques. The training was done with a *Tensorflow* [1] implementation of the *ADAM* [12] algorithm.

4.3 Datasets

To train the adversarial filter neural networks, both data and image classifiers were needed. The experiments conducted in this report were based on three datasets: The *MNIST* dataset [16] consisting of 70000 28×28 grayscaled images of handwritten digits between $0-9$, the *CIFAR-10* dataset [14] consisting of 65000 32×32×3 colored images belonging to 10 classes (airplanes birds, cats etc.) and the *Fashion-MNIST* dataset [32] consisting of 70000 28×28 grayscaled images of clothes belonging to 10 classes. When training filters as shown in Figure 4.2 the datasets were augmented with an equal amount of adversarial examples doubling the size of the datasets. The datasets were split into training and validation sets with a ratio of 4/1.

4.4 Constructing Adversarial Examples for the Filters

To generate adversarial examples used for training the adversarial filters the *CleverHans* [19] package in Python was used which among other things provides implementations of many adversarial attacks and benchmarks of the vulnerabilities to adversarial examples. Two attacks were chosen, the FGSM attack and the PGD attack. The FGSM attack was chosen for its simplicity and PGD attack was chosen since it is a stronger optimization based attack [6].

4.4.1 Generating FGSM Adversarial Examples

There are two parameters to consider when constructing an FGSM attack. The distortion size $||\delta||_\infty$ and whether or not the attack is targeted or not. In the experiments the adversarial examples were constructed with ℓ_∞-distances of 0.05 and 0.1 from the original images for both targeted and nontargeted attacks. In the case of targeted attacks, the target label was chosen randomly for each image.

4.4.2 Generating PGD Adversarial Examples

Similarly to the FGSM attack the distortion size $||\delta||_\infty$ and the target label are the parameters of choice when constructing the attack. The attacks were done with ℓ_∞-distances of 0.05 and 0.1 from the original images for both targeted and non-targeted attacks.

4.4.3 Adaptive Attacks

To further test the filters, two adaptive attacks were constructed. Since the filters are aimed to detect adversarial examples by detecting shifts in explainable AI heatmaps an attacker with knowledge of this defence would have two ways to attempt to fool the defence. Firstly, the attacker could construct an attack that aims to make the heatmaps of the adversarial examples as similar as possible to the heatmaps from the original images. Such an attack was created with the PGD attack seen in Equation (3.12) but adding the extra term $\lambda||\text{heat}(x) - \text{heat}(x')||_2$ to the objective where $\text{heat}(\cdot)$ is any explainable AI function. The additional term forces the attack to construct an image that does not alter the heatmaps as much as a normal attack would.

The second adaptive attack aims to construct images that are adversarial to the original classifier where their heatmaps are adversarial to the filter neural network too. In comparison to the first attack this attack does not aim to make the heatmaps of the adversarial examples identical to the heatmaps of the original data. The attack was constructed by adding the term $\lambda \log(y_l^{\text{filter}})$ to the objective of the PGD attack where y_l^{filter} is the output of the filter neural network corresponding to the probability of the input being original data.

4.5 Construction of Explainable AI Heatmaps

The generation of the explainable AI heatmaps was done with the help of public github repositories from the original creators of the methods. Two methods were chosen: Grad-CAM and layer-wise relevance propagation. The methods were chosen since they are specifically targeted for neural network image classifiers. The heatmaps were generated directly from Equations (3.34)– (3.35) and were scaled to be in the range of 0–1.

4.6 Models

The image classifier models used for the experiments were all standard convolutional neural networks followed by a fully connected layer. For MNIST and Fashion-MNIST we trained a three layered convolutional neural network reaching 99.1% and 91% accuracy respectively on test data and for the CIFAR dataset we trained a five layered convolutional neural network reaching 75% accuracy on test data. The filter neural networks were all two layered convolutional neural networks followed by a fully connected layer. The filter architectures are shown in Figure 4.3.

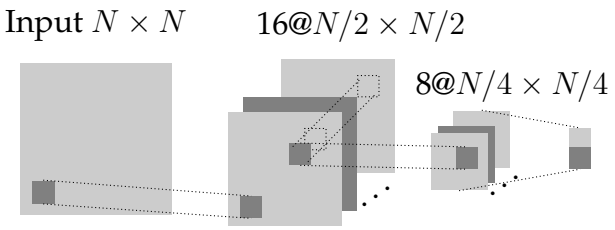


Figure 4.3: Filter neural network architecture. For the MNIST and Fashion-MNIST datasets $N = 28$ and for the Cifar dataset $N = 32$.

4.7 Evaluation Strategy

To objectively evaluate the effectiveness and robustness of the adversarial filters we have considered several metrics. Firstly, the accuracy of the filters was measured i.e., the fraction of the images correctly classified as adversarial or original. The accuracy was evaluated both on test data containing adversarial examples constructed with the same attack methods used in the training set and adversarial examples constructed using a different attack than the training set. This was to measure the robustness of the filters to ensure they are not only detecting features in data that are specific to the attack used for the training set. Lastly, the filters were evaluated on the adaptive attacks from Section 4.4.3 to measure to what extent an attacker with knowledge of the defences could bypass them.

Chapter 5

Results

This chapter will display the results obtained from the experiments.

5.1 Adversarial Filter Performance

In Tables 5.1– 5.3 the performances of the adversarial filters specified in Sections 4.2 and 4.6 are shown with respect to several parameters. The first column specifies the attack used to construct adversarial examples for the training data, the second column shows the inputs of the filters, the third column shows the distortion size $||\delta||_\infty$ of the adversarial examples and the last two column shows the accuracy of the filters when evaluating on adversarial data constructed with the PGD attack and FGSM attack respectively.

Table 5.1: MNIST adversarial filter performance.

Attack	**Input**	ε	**PGD-Acc**	**FGSM-Acc**
PGD	Images only	0.05	0.82	0.85
PGD	Images only	0.1	0.85	0.86
PGD	Grad-CAM	0.05	0.72	0.71
PGD	Grad-CAM	0.1	0.80	0.78
PGD	LRP	0.05	0.75	0.74
PGD	LRP	0.1	0.77	0.81
FGSM	Images only	0.05	0.79	0.85
FGSM	Images only	0.1	0.82	0.86
FGSM	Grad-CAM	0.05	0.70	0.75
FGSM	Grad-CAM	0.1	0.74	0.75
FGSM	LRP	0.05	0.75	0.70
FGSM	LRP	0.1	0.75	0.73

Table 5.2: Fashion-MNIST adversarial filter performance.

Attack	**Input**	ε	**PGD-Acc**	**FGSM-Acc**
PGD	Images only	0.05	0.94	0.96
PGD	Images only	0.1	0.96	0.95
PGD	Grad-CAM	0.05	0.82	0.88
PGD	Grad-CAM	0.1	0.90	0.89
PGD	LRP	0.05	0.85	0.92
PGD	LRP	0.1	0.90	0.91
FGSM	Images only	0.05	0.92	0.95
FGSM	Images only	0.1	0.94	0.94
FGSM	Grad-CAM	0.05	0.82	0.86
FGSM	Grad-CAM	0.1	0.83	0.89
FGSM	LRP	0.05	0.85	0.90
FGSM	LRP	0.1	0.86	0.93

Table 5.3: Cifar adversarial filter performance.

Attack	**Input**	ε	**PGD-Acc**	**FGSM-Acc**
PGD	Images only	0.05	0.91	0.94
PGD	Images only	0.1	0.94	0.90
PGD	Grad-CAM	0.05	0.82	0.88
PGD	Grad-CAM	0.1	0.90	0.89
PGD	LRP	0.05	0.87	0.93
PGD	LRP	0.1	0.92	0.86
FGSM	Images only	0.05	0.91	0.93
FGSM	Images only	0.1	0.94	0.95
FGSM	Grad-CAM	0.05	0.80	0.83
FGSM	Grad-CAM	0.1	0.85	0.89
FGSM	LRP	0.05	0.87	0.91
FGSM	LRP	0.1	0.82	0.94

5.2 Adaptive Attacks

Adaptive attacks were conducted on the filters. The second adaptive attack in Section 4.4.3 managed to fool all of the filters in Section 5.1 with a success rate of 100% when using $\lambda = 1$ for the additional term in the PGD attack.

In Figure 5.1 an adversarial example constructed with the first adaptive attack in Section 4.4.3 for $\lambda = 0.05$, an original image, and their Grad-Cam heatmaps are shown.

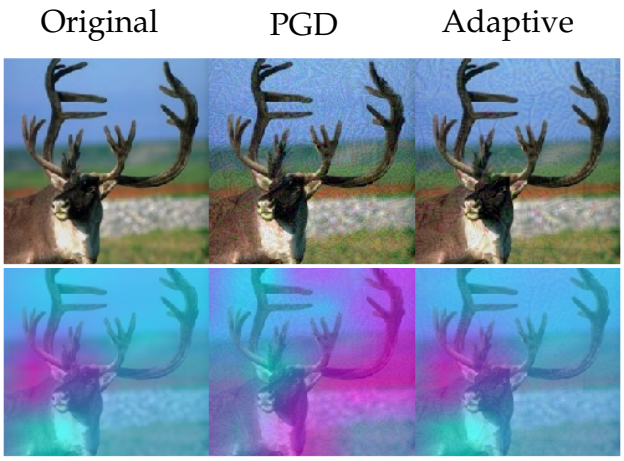


Figure 5.1: Comparison of heatmaps between a PGD adversarial example and an adaptive attack adversarial example. The original image is classified as an impala and both of the adversarial examples are classified as toilet tissue paper.

How the scalars of the adaptive attack evolve during the iterations of the ADAM algorithm is shown in Figure 5.2.

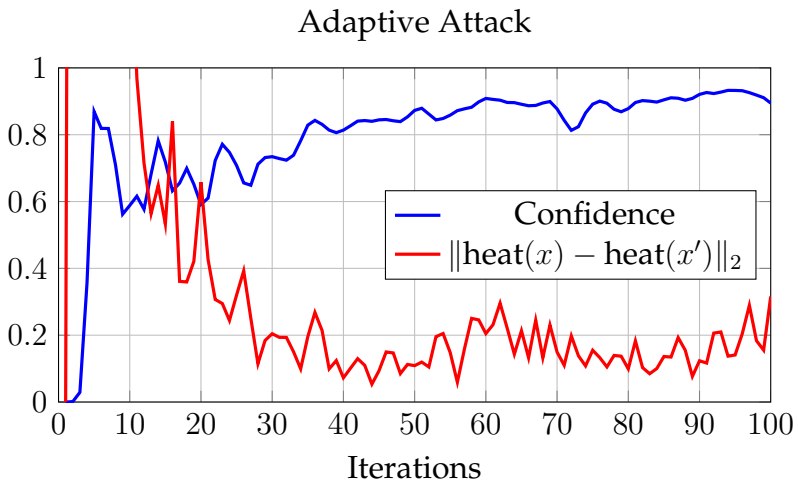


Figure 5.2: Probability of the target label and the ℓ_2 distance between the Grad-CAM heatmaps of the original image and the adversarial image against the number iterations of the attack.

Chapter 6

Discussion

This section will provide an analysis of the results obtained in Chapter 5. Pros and cons of the method of defending neural networks against adversarial examples using explainable AI will be highlighted in comparison to other state-of-the-art methods. Also the overall threat of adversarial examples with respect to the current defence methods will be discussed.

6.1 Threat of Adversarial Examples

Adversarial examples are not only a virtual threat. They can very well be constructed in the physical world and remain adversarial after being photographed, compressed and altered by many more transforms that occur in the real world. They can also be very general, meaning that they can remain adversarial to multiple entirely different neural network architectures. Even though the creation of such adversarial examples requires a more sophisticated attack algorithm they can be constructed without a significant computational cost.

Figure 3.1 shows an adversarial example constructed using the EOT attack targeted to be classified as a combination lock. Even when subjected to random transformations including rotation change, cropping, brightness change and additive Gaussian noise the image remains adversarial and gets classified as the target label. It is also seen in Figure 3.1 that a regular attack such as the PGD method does not retain the adversarial properties when subjected to the same transformations.

While the EOT attack highlights that adversarial examples can be constructed in the real world they are still quite limited as they can only be used for one purpose. A more effective attack would be to have an object that can be used classify any input to the neural networks as a wanted label. This is covered by the adversarial patch attack. Figures 3.2 and 3.3 shows nine different adversarial stickers. When placing the stickers on an image the image gets misclassified as the target label of the sticker. The stickers in the image were optimized to make any image classified incorrectly as the sticker's class by any neural network when placed on an arbitrary position on the image. Interestingly the more robust the stickers were made the more they started to look like the classes they were targeting. In Figure 3.2 one can clearly see the features of the target classes of each sticker. These features can also be hidden by disguising the stickers as seen in Figure 3.3.

6.2 Challenges in Using Correct Training Data

When training the adversarial filters we used inputs originating from correctly classified data and adversarial data. There is a challenge in capturing their true distributions and incorrect conclusions can be drawn when they are not sampled correctly. For instance, in the case of only using the data from the MNIST dataset and their adversarial examples as training data will yield an accuracy of the filters of approximately 100%. Unfortunately, this does not necessarily mean that the filters are learning features present in adversarial examples but could rather be that the filters, for instance, are discovering that the black background present in all of the MNIST images are shifted to a slightly grayer tone for the adversarial examples. The gray tone is not a defining feature of adversarial examples. To combat this we included failed attacks i.e., inputs that were still classified correctly after an attack were added to the dataset to force the filters to learn harder features. Adding failed attacks to the training set made the filters able to detect adversarial examples from more attacks but gives no guarantees that the filters are learning general adversarial features.

6.3 Is the Hypothesis True?

As shown in Tables 5.1– 5.3 the filter neural networks are capable of detecting adversarial examples for all of the input strategies. Even when evaluating the filters on data constructed by a different attack there was no significant drop in accuracy. The filters trained directly with images without heatmaps outperformed the filters trained using the various heatmap input strategies. This shows that the explainable AI heatmaps do not include further information regarding patterns in adversarial examples than the original inputs themselves. This is of no surprise, even though explainable AI would capture critical patterns there is nothing that speaks against the filters from identifying the same patterns seen in the heatmaps directly from the images themselves. However, in terms of human evaluation of the neural networks the heatmaps still are of great importance as they can in many cases pinpoint when neural networks do not work as intended. This is seen in Figure 3.5 where there is a notable shift in the Grad-CAM heatmap when an image has been distorted with an adversarial attack. In contrast, the shift is not as notable for the LRP heatmaps which aligns with the results by Bach et al. [4].

Both the performance of the filters and the shift in explainable AI metrics might point to that the hypothesis is true, that explainable AI metrics *can* be used to detect adversarial examples. However, even though normal attacks might be easily detectable with an adversarial filter an attacker can still with little effort bypass the filters with adaptive attacks (see Section 4.4.3). It is a slightly harder problem for the attacker to solve as there are more constraints that the attack must fulfill but in practice it is an easy attack to construct. Out of the filters used in this thesis, all of them failed to detect adversarial examples constructed using the second adaptive attack in Section 4.4.3. The attacks show a serious flaw of the method as it is easily bypassed when the attacker has knowledge of the defence mechanism.

One natural question to ask is whether or not transparency ever can be used to robustly tackle adversarial examples using the method proposed in this thesis. The method in the thesis is based on the assumption that an adversarial example x' causes a shift in the explainable AI metrics that can be captured with a filter. If a significant shift can

be caused with an adversarial example it means that the explainable AI technique is very sensitive to small changes of the input. This makes it likely that the explainable AI technique can also be manipulated as in Figure 5.2 making it possible for an attacker to bypass the method. Furthermore, if we assume the contrary, that adversarial examples do not cause a significant shift in the explainable AI heatmaps, the method proposed in the thesis would also not work. If adversarial examples do not cause a shift in the heatmaps then there is nothing for the filter to detect. This is a problem many proposed defence mechanisms have faced and is the main reason some researchers only advocates provable defence mechanisms [5] such as the convex adversarial defence in Section 3.5.3.

6.4 Difficulties in Achieving Robustness

As done in this thesis and many research papers the ℓ_p norm is used as a measure of proximity between original data and adversarial examples. There is no evidence that such a norm optimally resembles human perceptual similarity. Furthermore, to have an objective of making a neural network robust to all perturbations $\mathcal{S} = \{\delta : ||\delta||_p \leq \varepsilon\}$ only makes sense if $C^*(x) = C^*(x + \delta)\, \forall \delta \in \mathcal{S}$. There is nothing that speaks against two images x_1, x_2 of two separate classes that are close in terms of the distance $||x_1 - x_2||_p$, especially for classification tasks with many classes that are visually close. This speaks against the objective to forcefully train a neural network to be robust for all ℓ_p norms bounded by ε. Consider Figure 6.1. When training a neural network there will always be points of the input domain that are close to each other that gets classified as different classes giving room for the existence of adversarial examples. To get rid of this problem an additional "do not know" class is required which lies along the borders of all classes. However, the "do not know" class is very complex and can be hard to teach to a neural network.

The above points highlights the great difficulty of adversarial defence mechanisms as there are many aspects to consider. To achieve true robustness there must be knowledge of the true distribution of the classes in the input domain which is practically impossible to know.

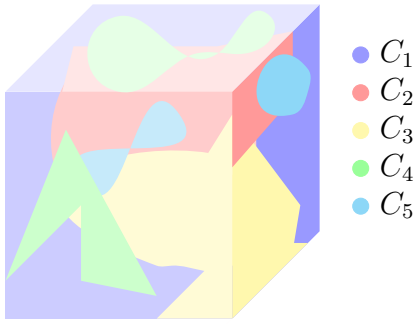


Figure 6.1: A visualization of how a neural network hypothetically would classify regions of the input domain.

6.5 Adversarial Defence Mechanisms in Practice

There is a conflict in whether or not neural networks should be defended with methods that are empirically shown to work or methods that provably work with respect to some metric. Some researchers argue that the provable defences are a necessity as there are no guarantees that defences that work empirically will not be broken by future attacks. However, the complexity of neural networks halts the practicality of implementing the provable defences present to this date for larger network architectures such as VGG16, resnet50 and inceptionv3. Even though progress has been made [30] the provable defences are still limited to networks aimed at the Cifar10 dataset scale.

Since there is such a vast range of perturbations that can cause neural networks to be fooled such as additive Gaussian noise, change in brightness and simple rotations Ford et al. [8] reasons that the existence of adversarial examples is a direct cause of the lack of robustness to image corruptions in general. This suggests that robustness to image corruptions implies adversarial robustness. In practice, this points to the adversarial retraining defence mechanism in Section 3.5.1 as the way to go to practically implement a solid defence. In contrast to the provable defences this will make it possible to robustify even neural networks that are very large as all the defence requires is an augmentation of the training data.

6.6 Ethics and Sustainability

The research of adversarial examples can seem a bit controversial as it provides information on many methods to harm neural networks. This information is accessible to anyone and will most likely draw the attention of people who seek to cause damage. There are countless research papers that show how to cause unwanted behavior from both image and audio classifiers. Even though there has been no major incident up to this point there are many indications that they will occur as deep neural networks gain more popularity. Since the attacks can be performed in a black box setting there is virtually nothing stopping attackers from causing harm. However, even without the research it would be inevitable that adversarial examples would be detected independently from the original discoverers. The open research also makes sure that the information of all of the threats to neural networks are openly accessible and allows neural network designers to take their own precautions. In the coming years the knowledge of the threat of adversarial examples will increase and will put a higher demand on research related to adversarial defence mechanisms.

Chapter 7

Conclusion

Explainable AI metrics have been used to train filter neural networks to detect adversarial examples. The objective was to investigate whether or not the explainable AI techniques could robustly extract patterns in adversarial examples that are not found in original data. The results show that the filters with layer-wise relevance propagation heatmaps as inputs can detect adversarial examples constructed with the projected gradient descent method with accuracies of 77%, 90%, and 92% for the MNIST, Fashion-MNIST, and CIFAR10 datasets respectively. However, the filters do not detect adversarial examples constructed with adaptive attacks that utilize the architecture of the defence mechanism. The results have provided evidence against the adversarial filters being robust enough to be trusted as a powerful defence. This thesis has highlighted the immense difficulty of defending neural networks against adversarial examples and shows that the proposed defence suffers from the same fragility against adaptive attacks as many proposed defences mechanisms already have.

7.1 Future Work

There are several areas in this field that need further research since many fundamental questions remain unanswered. First and foremost, do the results scale? This thesis has investigated filter neural networks on smaller neural network architectures and before drawing a definite conclusion of the research question the same tests should be done on larger, more complex networks.

Another question to investigate is whether or not it is possible to create a metric that resembles human perceptual similarity. With such a metric it will be easier to define how well an adversarial example is disguised. Moreover, it will be possible to investigate whether or not it is possible to construct adversarial examples that can fool humans as well. Most if not all adversarial attacks to this date are aimed to fool neural networks. If adversarial examples that fool humans exists the vital question would be to ask if they share similarities between adversarial examples constructed to fool neural networks.

In Theorem 1 it was shown that the possibility of ensuring robustness is limited to the properties of the dataset that the network is trained to classify. It is likely that there are additional limitations not only on the dataset but also on the network architecture and the dimensionality of the input that further restricts the possibility of robustness. Knowledge of these limitations are critical in the future search of robust defence mechanisms.

As most of the defence mechanisms to this date aim to defend against adversarial examples that are similar to the original inputs there is a need to further extend defence mechanisms to adversarial examples that contain larger distortions, like an adversarial patch. Adversarial examples with large distortions are in many cases even more dangerous than regular adversarial examples and should therefore be considered more. However before this can be addressed properly there must exist a solid robust defence which currently does not exist.

Bibliography

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. "TensorFlow: A system for large-scale machine learning". In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. 2016, pp. 265–283.

[2] Anish Athalye, Nicholas Carlini, and David A. Wagner. "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples". In: *CoRR* abs/1802.00420 (2018). arXiv: 1802.00420.

[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. "Synthesizing Robust Adversarial Examples". In: *CoRR* abs/1707.07397 (2017). arXiv: 1707.07397.

[4] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation". In: *PloS one* 10.7 (2015), e0130140.

[5] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian J. Goodfellow, Aleksander Madry, and Alexey Kurakin. "On Evaluating Adversarial Robustness". In: *CoRR* abs/1902.06705 (2019). arXiv: 1902.06705.

[6] Nicholas Carlini and David Wagner. "Towards evaluating the robustness of neural networks". In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2017, pp. 39–57.

[7] Nicholas Carlini and David A. Wagner. "Defensive Distillation is Not Robust to Adversarial Examples". In: *CoRR* abs/1607.04311 (2016). arXiv: 1607.04311.

[8] Nic Ford, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk. "Adversarial Examples Are a Natural Consequence of Test Error in Noise". In: *CoRR* abs/1901.10513 (2019). arXiv: 1901.10513.

[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[10] David Gunning. "Explainable artificial intelligence (xai)". In: *Defense Advanced Research Projects Agency (DARPA), nd Web* (2017).

[11] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. "Deep speech: Scaling up end-to-end speech recognition". In: *arXiv preprint arXiv:1412.5567* (2014).

[12] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[13] J Zico Kolter and Eric Wong. "Provable defenses against adversarial examples via the convex outer adversarial polytope". In: *arXiv preprint arXiv:1711.00851* 1.2 (2017), p. 3.

[14] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. *CIFAR-10 (Canadian Institute for Advanced Research)*. URL: http://www.cs.toronto.edu/~kriz/cifar.html.

[15] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. "Adversarial Machine Learning at Scale". In: *CoRR* abs/1611.01236 (2016). arXiv: 1611.01236.

[16] Yann LeCun and Corinna Cortes. *MNIST handwritten digit database*. 2010. URL: http://yann.lecun.com/exdb/mnist/.

[17] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. "Delving into Transferable Adversarial Examples and Black-box Attacks". In: *CoRR* abs/1611.02770 (2016). arXiv: 1611.02770.

[18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. "Towards deep learning models resistant to adversarial attacks". In: *arXiv preprint arXiv:1706.06083* (2017).

[19] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. "Technical Report on the CleverHans v2.1.0 Adversarial Examples Library". In: *arXiv preprint arXiv:1610.00768* (2018).

[20] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection". In: *CoRR* abs/1506.02640 (2015). arXiv: 1506.02640.

[21] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[22] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. "Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization". In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[23] Ali Shafahi, W. Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. "Are adversarial examples inevitable?" In: *CoRR* abs/1809.02104 (2018). arXiv: 1809.02104.

[24] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps". In: *CoRR* abs/1312.6034 (2013). arXiv: 1312.6034.

[25] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *CoRR* abs/1409.1556 (2014). arXiv: 1409.1556.

[26] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*. 2014, pp. 3104–3112.

[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. "Intriguing properties of neural networks". In: *CoRR* abs/1312.6199 (2013). arXiv: 1312.6199.

[28] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[29] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. "Ensemble adversarial training: Attacks and defenses". In: *arXiv preprint arXiv:1705.07204* (2017).

[30] Eric Wong, Frank R. Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. "Scaling provable adversarial defenses". In: *CoRR* abs/1805.12514 (2018). arXiv: 1805.12514.

[31] Shangxi Wu, Jitao Sang, Kaiyuan Xu, Jiaming Zhang, Yanfeng Sun, Liping Jing, and Jian Yu. "Attention, Please! Adversarial Defense via Attention Rectification and Preservation". In: *CoRR* abs/1811.09831 (2018). arXiv: 1811.09831.

[32] Han Xiao, Kashif Rasul, and Roland Vollgraf. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. Aug. 28, 2017. arXiv: cs.LG/1708.07747 [cs.LG].

[33] Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. "Adversarial Examples: Attacks and Defenses for Deep Learning". In: *CoRR* abs/1712.07107 (2017). arXiv: 1712.07107.

[34] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. "Learning Deep Features for Discriminative Localization". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.

[35] Jacek M Zurada. *Introduction to artificial neural systems*. Vol. 8. West publishing company St. Paul, 1992.

TRITA -EECS-EX:466