KU LEUVEN

FACULTY OF
ENGINEERING SCIENCE

Distributed Adversarial Attacks

Sander Prenen

Thesis voorgedragen tot het behalen
van de graad van Master of Science
in de ingenieurswetenschappen:
computerwetenschappen, hoofdoptie
Artificiële intelligentie

Promotoren:
Prof. dr. ir. W. Joosen
Dr. ir. D. Preuveneers

Assessoren:
Ir. W. Eetveel
W. Eetrest

Begeleiders:
V. Rimmer
I. Tsingenopoulos

Academiejaar 2021 – 2022

Preface

Sander Prenen

Contents

List of Figures

List of Tables

List of Abbreviations

AI Artificial Intelligence. 3

ANN Artificial Neural Network. 3, 5

API Application Programming Interface. 5

CNN Convolutional Neural Network. 3

DNN Deep Neural Network. 3

EA Evolutionary Algorithm. 7

FGSM Fast Gradient Sign Method. 5

PSO Particle Swarm Optimization. 7

ReLU Rectified Linear Unit. 3, 4

RNN Recurrent Neural Network. 3

Chapter 1

Introduction

Chapter 2

Background

2.1 Neural networks

Ever since the invention of computer systems, it has always been a goal of scientists and engineers to create Artificial Intelligence (AI). Current state of the art approaches are mimicking the human brain, more specifically the neurons inside the brain. Already in the fifties, Rosenblatt introduced his perceptron [1]. The perceptron is a single neuron able to learn linearly separable patterns. It does so by finding a hyperplane that separates the two classes. This hyperplane is called the decision surface or decision boundary and the perceptron itself is called a classifier. Geometric regions separated by a decision boundary are called decision regions. The concept of linear separability is explained in Figure 2.1 in two dimensions. In Figure 2.2, the decision boundaries and decision regions are explained visually.

Unfortunately not all patterns are linearly separable. To overcome this problem, the neurons can be layered, creating an Artificial Neural Network (ANN) in the process. Layering neurons sequentially is essentially a linear combination of neurons. This in itself does not create non-linear decision surfaces. Non-linear activation functions are added for the ANN to be able to learn more complex decision boundaries. Some commonly used activation functions are Rectified Linear Unit (ReLU) [2], Heaviside step function and softmax (or sigmoid when used on scalars). In Figure 2.3 the plots of the activation functions can be found.

The neurons can be combined in different ways to create different ANN architectures. Each architecture has its own strengths and weaknesses. Convolutional Neural Networks (CNNs) excel in classifying visual data [3, 4], whilst Recurrent Neural Networks (RNNs) are widely used when there exist dependencies inside the data, such as in speech recognition [5, 6] or time series prediction [7]. More recent research focuses on Deep Neural Networks (DNNs), due to the ever increasing computational power available. DNN approaches are able to compare to and even surpass human performance [8, 9].

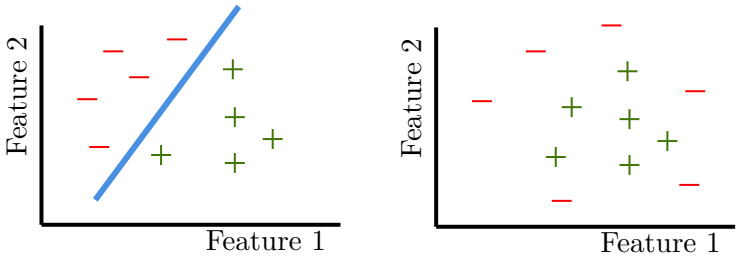

Figure 2.1: Linearly separable classes on the left and non-linearly separable classes on the right. Two classes are linearly separable if there exists a hyperplane for which all examples of one class are on the same side of this hyperplane, whilst all examples of the other class are on the other side of the hyperplane. In two dimensions, the hyperplane is a straight line.

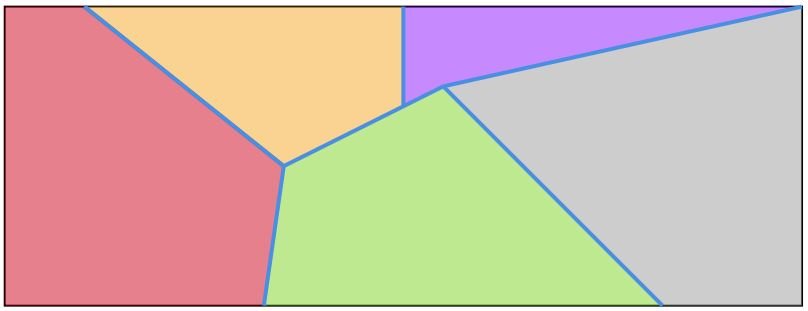

Figure 2.2: Decision boundaries (blue lines) separate different decision regions (colored regions).

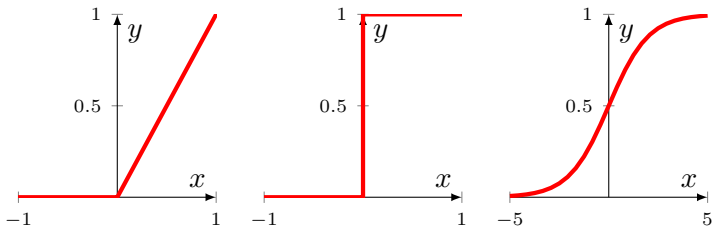

Figure 2.3: Plots of different activation functions. From left to right: ReLU, Heaviside step and sigmoid.

2.2 Adversarial attacks

The expressiveness of ANNs is a double-edged sword. It is the cause for the near-human performance on some tasks, but also for counter-intuitive properties. As studied by Szegedy et al [10], one of these properties is the presence of discontinuous decision boundaries. This might cause seemingly identically images to be classified differently. They first defined adversarial examples as *imperceptibly small perturbations to a correctly classified input image, so that it is no longer classified correctly* [10]. This property of ANNs might not seem important at first glance, but it can be quite worrisome from a security point-of-view. Malicious users could craft images to bypass face recognition software [11] or attack the camera of a self-driving car to misclassify traffic signs [12]. Other fields where adversarial examples are of interest include malware detection [13], natural language processing [14] and industrial control systems [15]. Adversarial attacks are algorithms used to craft such adversarial examples.

Most research on adversarial attacks is done using images. Researchers have the most freedom in this domain, since a slightly altered image is still an image with roughly the same contents. Slightly modifying an industrial control system however, might break the entire way the system works. Research in other domains is mostly conducted by altering existing image algorithms to the specific use case. For this reason this works only focuses on adversarial attacks on images.

2.2.1 Adversarial attacks terminology

Adversarial attacks are generally divided in two categories, white box attacks and black box attacks. In a white box attack, the attacker has complete knowledge of the classifier under attack. This knowledge consists of the architecture, parameters and thus their gradients and all output of the classifier. Examples of white box attacks are the Fast Gradient Sign Method (FGSM) [16] or the Carlini & Wagner attack [17].

In black box attacks, the only thing the attacker has access to is the output of the model. Depending on the literature, this output consists of class labels only (decision-based attack) or class labels and the corresponding confidence scores (score-based attacks). Black box attacks are more relevant in real-life scenarios, since most attacks are performed on a third-party Application Programming Interface (API). These APIs generally do not reveal the underlying model.

Both white box and black box attacks can be divided into targeted and untargeted attacks depending on their goal. In a targeted attack, the goal of the attacker is to create an adversarial example with a specific target class. In an untargeted attack the target class can be any class. Untargeted variants of attacks generally enjoy much more freedom and are therefore able to craft adversarial examples that are

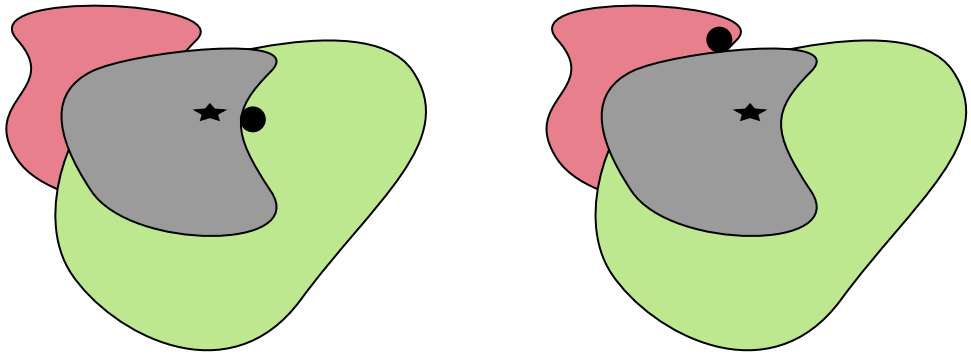

Figure 2.4: Different decision regions are shown in different colors. An adversarial example is being created starting from the source image (black star). On the left an untargeted attack is performed. The adversarial example is the image closest to the source image, that is classified differently (black circle). On the right a targeted attack is performed with the red decision region being the target class. The adversarial example is the image closest to the source image that is in the red decision region.

closer to the original. Figure 2.4 visually explains the difference between the two types.

What does it mean for images to be close to each other? This is easy to visualize in two dimensions as in Figure 2.4, but in higher dimensions, this is more difficult. Images reside in d-dimensional space, where d is the amount of pixels of the image. Two commonly used distances in higher dimensions are the L_2-distance and the L_∞-distance. They are defined as follows:

$$
\begin{aligned}
L_2(X,Y) &= \sqrt{\sum_{i=1}^{d} |x_i - y_i|^2} \\
L_\infty(X,Y) &= \lim_{p \leftarrow \infty} \left(\sum_{i=1}^{d} |x_i - y_i|^p \right)^{1/p} \\
&= \max(|x_1 - y_1|, |x_2 - y_2|, \ldots, |x_d - y_d|)
\end{aligned}
$$

In both distances X and Y represent the images and $(x_1, x_2, \ldots, x_i, \ldots, x_d)$ and $(y_1, y_2, \ldots, y_i, \ldots, y_d)$ are the pixel values of X and Y respectively. The L_2-distance is also known as the Euclidean distance, which is a generalization of the Pythagorean theorem in more than two dimensions. It takes the pairwise distances between all pixels into account. The L_∞-distance is also called the Chebyshev distance. This distance only depends on the maximal pairwise distance between the two images. By minimizing the L_∞-distance, the maximal pixelwise difference in minimized [18].

2.3 Particle swarm optimization

Particle Swarm Optimization (PSO) is an optimization framework part of the Evolutionary Algorithms (EAs) family. In EAs, populations of candidate solutions evolve based on mechanisms inspired by the field op biology, such as ant colonies [19], mutation and recombination [20]. The mechanism that inspired PSO is the flocking of birds.

Bibliography

[1] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. 65(6):386–408.

[2] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress.

[3] M.V. Valueva, N.N. Nagornov, P.A. Lyakhov, G.V. Valuev, and N.I. Chervyakov. Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. *Mathematics and Computers in Simulation*, 177:232–243, 2020.

[4] Steve Lawrence, C. Lee Giles, Ah Chung Tsoi, and Andrew D. Back. Face recognition: A convolutional neural network approach. *IEEE TRANSACTIONS ON NEURAL NETWORKS*, 8(1):98–113, 1997.

[5] Xiangang Li and Xihong Wu. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. *CoRR*, abs/1410.4281, 2014.

[6] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[7] Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. Recurrent neural networks for time series forecasting: Current status and future directions. *CoRR*, abs/1909.00590, 2019.

[8] Jon Russell. Google's alphago ai wins three-match series against the world's best go player. https://techcrunch.com/2017/05/24/alphago-beats-planets-best-human-go-player-ke-jie/amp/?guccounter=1, 05 2017. Accessed: 2021-12-08.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[10] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014.

[11] Fatemeh Vakhshiteh, Ahmad Nickabadi, and Raghavendra Ramachandra. Adversarial attacks against face recognition: A comprehensive study, 2021.

[12] Abhiram Gnanasambandam, Alex M. Sherman, and Stanley H. Chan. Optical adversarial attack, 2021.

[13] Octavian Suciu, Scott E. Coull, and Jeffrey Johns. Exploring adversarial examples in malware detection, 2019.

[14] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text, 2018.

[15] Giulio Zizzo, Chris Hankin, Sergio Maffeis, and Kevin Jones. Invited: Adversarial machine learning beyond the image domain. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–4, 2019.

[16] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[17] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks, 2017.

[18] Distance. Distance. https://en.wikipedia.org/wiki/Distance, 11 2021. Accessed: 2021-12-09.

[19] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system: Optimization by a colony of cooperating agents. ieee trans syst man cybernetics - part b. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 26:29–41, 02 1996.

[20] John H. Holland. *Genetic Algorithms and Adaptation*, pages 317–333. Springer US, Boston, MA, 1984.