UNIVERSITY OF PISA

DEPARTMENT OF COMPUTER SCIENCE

Master's Degree in Computer Science
AI curriculum

Intelligent Systems for Pattern Recognition:

Adversarial Attacks:

Review of the main approaches to perform adversarial attacks to neural networks

Prof:

Bacciu Davide

Candidate:

Paterniti Barbino Niko
Mat: 638257

ANNO ACCADEMICO 2022/2023

Indice

Capitolo 1

Introduction

Deep learning models have demonstrated remarkable success across various domains, yet their vulnerability to adversarial attacks poses significant security concerns. This study presents a comprehensive review and analysis of adversarial attacks and defenses in the context of different deep learning paradigms.
Recent research has shown that machine learning models, including deep neural networks (DNNs), excel in pattern recognition tasks like image classification, even reaching human-level performance in tasks such as DeepFace. However,these models can be manipulated into misclassifying images with high confidence by adding imperceptible perturbations. These adversarial images pose a serious threat to machine learning systems, potentially leading to security breaches in various applications like ATMs, surveillance systems, face recognition, speech recognition, and even AI assistants like Apple Siri, Amazon Alexa, and Microsoft Cortana.
Firstly, we will show some of the most common approaches used to perform adversarial attacks, exploring the susceptibility of deep neural networks (DNNs) to adversarial examples and showcasing their ease of manipulation by adversaries. We will see a consistent drop in the performances of the algortihms even with small perturbations that remain imperceptible to human perception.
Then, various defense mechanisms aimed at mitigating adversarial threats will be presented, from the addition of adversarial training data to the analysis of defensive mechanisms based on Generative Adversarial Network(GAN).
The analysis enlightens the critical need for robust defense mechanisms to safeguard deep learning systems against adversarial attacks, emphasizing the importance of ongoing research in this rapidly evolving field.
The methods presented offer concrete security measures for training and attacking neural networks, providing insights into the guarantees they offer. Importantly, they demonstrate resistance to various attacks, not just specific ones, thus offering robust protection against potential attacks.

Capitolo 2

Adversarial Attacks

The typical goal of an adversarial attack is to introduce subtle perturbations, like changes in lighting or adding noise, to an image so that a machine learning model misclassifies it while remaining visually unchanged to humans. There are three main types of adversarial attacks:

- Non-targeted adversarial attacks: These aim to deceive a classifier by altering the source image, causing the model to output any random class other than the correct one.
- Targeted adversarial attacks: These are tailored to misclassify an image as a specific target class by modifying the source image. For example, an adversarial image might be crafted to impersonate a specific individual or object.
- Defenses against adversarial attacks: These focus on developing robust classifiers capable of correctly identifying adversarial images.

2.1 Fast Gradient Sign Method(FGSM)

The Fast Gradient Sign Method (FGSM) and its variations are practical approaches for generating adversarial examples in machine learning models, particularly neural networks.These methods take advantage of the linear behavior of models to efficiently perturb input data and cause misclassification.
By exploiting the linear behavior of models and employing a simple gradient-based approach, FGSM quickly generates adversarial examples(a sample input data modified to cause a model to incorrectly classify it) that reliably lead to misclassification. FGSM operates by computing the gradient of a loss function $J(\theta, x, y)$ with respect to the input data x. The perturbation η is then added to the input data in the direction that maximizes the loss:

$$\eta = \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$$

Where:

- $J(\theta, x, y)$ represents the cost function for training a neural network,
- η denotes the perturbation,
- θ signifies the parameters of the model,
- x denotes the input,
- y represents the target label, if applicable,
- ϵ is a small constant determining the magnitude of the perturbation.

The perturbation η is scaled by ϵ and added to the original input data x, resulting in an adversarial example x'. This adversarial example is utilized to cause misclassification by the neural network while remaining imperceptible to human observers. The main idea behind FGSM is to add some weak noise on every step of optimization that moves towards the expected class or away from the correct one. To keep the attack subtle, the amplitude (intensity of pixel channel) of noise has to be limited. This noise optimization is akin to adjusting for error, with the gradient guiding the direction of perturbation. In practical scenarios, FGSM can be used to attack black box models, where only the input data and output class are accessible. The process involves generating noise and adding it to the image iteratively until the neural network misclassifies it. Despite variations in the amplitude of the noise, the true class is never returned. A simple binary search helps determine the weakest noise that consistently leads to misclassification.
The equation

$$w^T x' = w^T x + w^T \eta$$

illustrates how the perturbation η affects the dot product between the weight vector w and the input data x. This demonstrates that a basic linear model can provide adversarial samples if there is enough dimensionality in its inputs. FGSM effectively moves in the direction towards the link between the misclassified result and the correct class. When the estimated class changes by following the gradient, it indicates the attack achieves successful results.

2.1.1 FGSM with Momentum

A variation of FGSM introduces momentum to the attack, making it more iterative. This momentum-based approach involves iteratively updating the perturbation to gradually steer the input data towards misclassification. The iterative process enhances the transferability of the attack across different models and datasets, while maintaining its effectiveness in causing misclassification. The update rule for the perturbation with momentum η_t at time step t is defined as:

$$\eta_{t+1} = \mu \cdot \eta_t + \frac{\nabla_x J_\theta(x'_t, l)}{\|\nabla_x J_\theta(x'_t, l)\|_2} \qquad x'_{t+1} = x'_t + \epsilon \cdot \mathrm{sign}(\eta_{t+1})$$

- μ is the momentum parameter,
- η_t represents the perturbation at time step t,
- x'_t denotes the adversarial example at time step t,
- $J_\theta(x'_t, l)$ is the loss function with respect to the adversarial example x'_t and its true label l,
- $\|\cdot\|_2$ denotes the L_2 norm.

FGSM with momentum enhances the transferability of the attack across different models and datasets while maintaining its effectiveness in causing misclassification. By iteratively updating the perturbation with momentum, FGSM with momentum explores a larger portion of the loss landscape and leads to stronger attacks compared to standard FGSM.

2.2 Projected Gradient Descent(PGD)

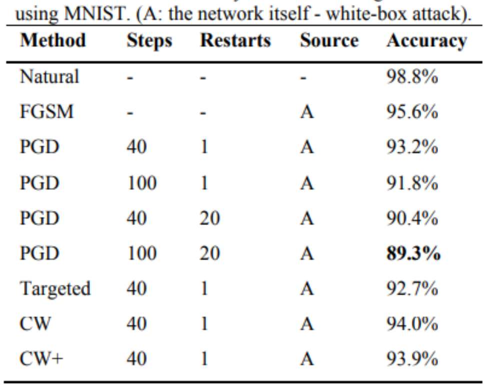
The multi-step variant of FGSM, also known as Projected Gradient Descent (PGD) is a powerful adversarial attack method that iteratively perturbs input data to maximize the loss function, with the constraint that the perturbed data remains within a specified distance from the original data. It is expressed as:

$$x_{t+1} = \pi_x \left(x_t + S\left(x_t + \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))\right)\right)$$

- x_t represents the adversarial example at iteration t,
- π_x denotes the projection function to ensure the perturbed data remains within a specified distance from the original data,
- S represents the step size of the optimization process,
- $J(\theta, x, y)$ is the loss function with respect to the adversarial example x_t and its true label y,
- ϵ is the magnitude of the perturbation.

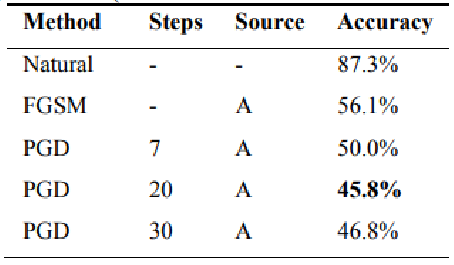
PGD is considered a stronger adversary compared to the FGSM method due to its iterative nature and ability to explore a larger portion of the loss landscape. FGSM, being a one-step approach, may lead to overfitting and lack of robustness,in fact the network might overfit to the adversarial examples generated by FGSM(label leaking). Robustness against PGD adversaries provides robustness against all first-order adversaries, making PGD a potent attack relying solely on first-order information. It is particularly effective for large-scale constrained optimization problems and can explore different local maxima in the loss landscape. In practice, PGD is used to train adversarially robust networks by solving the outer optimization problem of the saddle point formulation. This involves finding model parameters that minimize the adversarial loss, representing the inner maximization problem. Empirical studies, such as those presented in Table 1 and Table 2, demonstrate the effectiveness of PGD as the best attack method, even against advanced network architectures like ResNet. Networks trained to withstand PGD adversaries show resilience against a wide range of attacks, indicating the importance of robust optimization techniques.

Table 1. Performance results of the adversarially trained model against each adversary for ε = 0.3 using MNIST. (A: the network itself - white-box attack).



Method	Steps	Restarts	Source	Accuracy
Natural	-	-	-	98.8%
FGSM	-	-	A	95.6%
PGD	40	1	A	93.2%
PGD	100	1	A	91.8%
PGD	40	20	A	90.4%
PGD	100	20	A	**89.3%**
Targeted	40	1	A	92.7%
CW	40	1	A	94.0%
CW+	40	1	A	93.9%

Table 2. Performance results of the adversarially trained model against each adversary for ε = 8 using CIFAR10. (A: the network itself - white-box attack).



Method	Steps	Source	Accuracy
Natural	-	-	87.3%
FGSM	-	A	56.1%
PGD	7	A	50.0%
PGD	20	A	**45.8%**
PGD	30	A	46.8%

Attacks like PGD, lack direct access to the target network, making them what we call zero-order attacks. They operate blindly, computing examples without gradient feedback making the task more difficult. If a network is trained to be strong against PGD adversaries, it can also be secure against a wide range of different attacks as well. The most common method for training deep learning models is the Stochastic Gradient Descent (SGD), which optimized the parameters of neurons each layer. By using methods like PGD, we can effectively address the inner optimization problem of minimizing the adversarial loss which quantifies the network's vulnerability to attacks. But in order to train an adversarial robust network, we also have to solve the outer optimization problem of the saddle point formulation.In other words, model parameters have to be found in order to minimize the adversarial loss, this involves fine-tuning the model's parameters further, ensuring that it remains resilient to potential threats.

2.3 Basic Iterative Method(BIM)

The Basic Iterative Method (BIM) is an iterative version of the Fast Gradient Sign Method (FGSM)s. Unlike FGSM, which applies adversarial noise only once, BIM applies adversarial noise iteratively, providing more control over the attack process. This method has been demonstrated to be more effective than FGSM, particularly on datasets like ImageNet. In BIM, adversarial noise is applied multiple times with a small step size. The process involves clipping pixel values to limit big changes on each pixel in every iteration, preventing drastic alterations to the input image. The iterative process of BIM is represented by the following equation:

$$x_{n+1} = \text{Clip}_{x,\Delta}(x_n + \epsilon \cdot \text{sign}(\nabla_x J(x_n, y)))$$

Where:

- x_n represents the adversarial example at iteration n,
- ϵ is the magnitude of the perturbation,
- $J(x_n, y)$ is the loss function with respect to the adversarial example x_n and its true label y,
- $\text{Clip}_{x,\Delta}(x')$ denotes clipping the pixel values of x' within a specified range Δ.

For adversarial training, the core idea is to inject adversarial examples into the training set continuously, creating new adversarial samples at each step of training. Initially, smaller models were used for adversarial training without batch normalization. However, batch normalization has been found crucial, especially when scaling adversarial training to large datasets like ImageNet. A loss function is typically employed to control the number and relative weight of adversarial examples in each training batch. Adversarial training using any type of one-step method has been shown to increase robustness against all types of one-step adversarial examples. However, the accuracy on clean and adversarial examples may vary depending on the combination of training and evaluation methods used. Regularization methods such as dropout, label smoothing, and weight decay can affect the performance of clean and adversarial examples differently. For example, adding a regularizer to adversarial training may increase accuracy on adversarial examples while decreasing accuracy on clean images. Overall, adversarial training is recommended in scenarios where overfitting is a concern and regularization is required, as well as in situations where security against adversarial examples is a priority. Despite potentially sacrificing a small amount of accuracy, adversarial training offers robust defense against adversarial attacks, making it a valuable technique for securing neural networks.

2.4 Carlini-Wagner(CW)

Carlini-Wagner (CW) is an approach capable of overcoming defensive distillation, a promising defense algorithm. Defensive distillation involves training a new neural network, known as the "distilled network," using a softened version of the original network's outputs as labels during training. This softened version is obtained by applying the softmax function with a higher temperature parameter to the logits (pre-softmax outputs) of the original network. The distilled network is trained to predict these softened labels, effectively smoothing out the decision boundaries learned by the original network. The intuition behind defensive distillation is that by training the distilled network to imitate the behavior of the original network at a higher temperature, it learns to generalize better and becomes less sensitive to small perturbations in the input space, such as those introduced by adversarial attacks. However, Carlini and Wagner demonstrated that defensive distillation is not effective against their proposed attacks, including the L2, L0, and L∞ attacks. These attacks were able to find adversarial examples with high success rates even against networks trained with defensive distillation, highlighting the limitations of this defense mechanism in practice. These attacks demonstrate a 100% success rate in generating adversarial samples causing misclassification with the same label, thereby defeating defensive distillation with 100% probability.. The process involves four steps:

1. Train a neural network.
2. Calculate the softmax in a smoother way and compute the soft training labels by applying the network to each case in the training dataset.
3. Train the distilled network with soft training labels.
4. Test the distilled network on new inputs to classify.

One of the main contributions of CW's work is that the L0 attack is the first approach resulting in specific misclassification on ImageNet. Furthermore, after applying all three attacks to defensive distillation, it is observed that there is little difference in security between undistilled and distilled networks. L0, L2, and L_∞ attacks have different ways in which the "distance" between the original input x and the manipulated input x' is calculated.

- The L0 attack measures distance by counting the number of pixels that have been changed between the original input and the manipulated input. In other words, it counts the number of modifications required to transform the original input into the manipulated one.
- The L2 attack measures distance by calculating the square root of the sum of the squares of the differences between each pixel of the original input and the

manipulated input. This approach considers the "similarity" between inputs based on their Euclidean distance in feature space.

- The L_∞ attack measures distance by taking the maximum of the absolute differences between each pixel of the original input and the manipulated input. Essentially, it evaluates the maximum possible change for a single pixel.

When comparing these attacks, the L2 attack is often preferred because it tends to produce manipulations with low distortion relative to the original input, meaning the manipulated input remains as similar as possible to the original. However, the L0 attack is known to be non-differentiable and therefore difficult to optimize using gradient descent techniques. Similarly, the L_∞ attack poses similar challenges, as its non-complete differentiability makes it difficult to use gradient descent for optimization.

2.5 Jacobian Based Saliency Map Approach(JSMA)

JSMA iteratively perturbs features of input data that have large adversarial saliency scores. This score describes the adversary's intent by moving a sample away from its source class towards a certain target class. While other methods use output variations to find related input perturbations for adversarial samples, JSMA constructs a mapping from input perturbations to output variations, known as the forward derivative. This is represented by the Jacobian matrix of the function learned by the DNN:

$$\nabla J(X) = \frac{\partial J(X)}{\partial X} = \left[\frac{\partial J_j(X)}{\partial x_i}\right]_{i=1..M, j=1..N}$$

In this formulation, X is the sample, $J(X)$ is the classified result by the network, M is the input dimension or the number of neurons in the input layer, and N is the output dimension or the number of neurons in the output layer. The Jacobian matrix represents the sensitivity of the output with respect to changes in the input. JSMA achieves a remarkable success rate of 97.10% in generating adversarial samples by perturbing only 4.02% of input features per image. The saliency map is a tool to visualize which input features should be modified to influence the network's output and exploiting the derivative of the neural network, JSMA seeks to find perturbations that lead to misclassification, specifically targeting a desired output class while minimizing changes to other classes.
Given an input y and a neural network M, JSMA aims to increase the probability $M(y)$ for a target class p while decreasing probabilities for all other classes. It achieves this by iteratively perturbing input features until the desired misclassification occurs. This iterative approach continues until the target misclassification is achieved, creating adversarial examples with minimal modifications.

Capitolo 3

Defensive Stratiegies

While DNNs leverage large datasets to map obscure relationships, they remain susceptible to adversarial attacks. Small perturbations applied to inputs to a network can achieve undesirable results by reducing its accuracy.

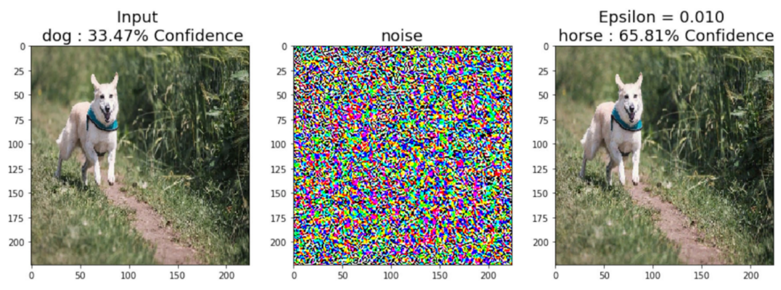


Fig. 1 With a small amount of perturbation, a deep neural network misclassifies a *dog* for a *horse*. The correct label (leftmost image) was predicted with a 33% confidence, whereas the wrong classification (rightmost) with a 66% probability

We can take as example a deep learning model based on the ResNet-50 architecture which performed a classification task achieving 97% accuracy on the validation set.Yet,its prediction accuracy against adversarial samples dropped to 10% with slight perturbation of the input image suggesting that even achieving state-of-the-art accuracy, there is an underlying flaw in how the network maps a given input to the predicted label.

3.1 Adversarial Training

The Adversarial Training method enhances the robustness of a model by injecting small poisonous samples in the training data to help the network reinforce its decision boundaries. The training set is augmented with counterexamples and their accompanying correct labels (x',y) so that the learning algorithm will correctly classify those inputs. However, despite the model being trained with adversarial examples, it may still exhibit vulnerabilities when faced with new, unseen attacks, especially those crafted using techniques that don't rely on detailed knowledge of the model's internals(black box attacks).

3.2 Defence-GAN

The Defence-Gan method is an original approach to defense that scales to both white-box and black-box attacks. The Defense-GAN method begins by training a generative adversarial network (GAN). A GAN consists of two neural networks: a generator and a discriminator. The generator learns to generate realistic data samples (e.g., images) from random noise, while the discriminator learns to distinguish between real data samples and those generated by the generator. Once the GAN is trained, it is used to generate adversarial examples. Adversarial examples are crafted by perturbing legitimate input data in such a way that it causes misclassification or errors in the target model. These perturbations are often imperceptible to human observers but can fool the machine learning model into making incorrect predictions.The adversarial examples generated by the GAN are then added to the training dataset of the target model. The model is retrained on this augmented dataset, which now includes both original and adversarial examples. By exposing the model to these adversarial samples during training, it learns to better distinguish between legitimate and adversarial inputs, thus improving its robustness against adversarial attacks.After retraining the model with the augmented dataset, it is evaluated on a separate validation or test dataset to assess its performance against adversarial attacks. Fine-tuning and additional adjustments may be performed based on the evaluation results to further enhance the model's robustness.

Wasserstein (WGAN) is an extension which makes the use of Wasserstein distance as a more efficient loss function to the original GAN.

$$\mathbb{E}_{x\sim p_{\text{data}}}\left[\min_{z}\|G_t(z)-x\|_2\right]\to 0 \tag{3.1}$$

During training, the generator G learns to generate outputs that fall within the data distribution of a benign input x in the form of G(z) where z is a random vector and the discriminator D discriminates between actual and generated samples. As defined by the equation, x is projected on G's range to minimize reconstruction loss at time step t. G and D are trained alternately. The reconstruction of G given the input z is then fed to the classifier. This approach generalizes to any classifier as the reconstruction layer is independent from underlying classification task.

3.3 Experiments

The experiments have been conducted on 170,000 perturbed images created using 3 different attack methods: JSMA, FGSM, CW and comparing the two defense strategies. The networks were trained on three datasets: CIFAR10 ,Fashion-MNIST (FMNIST) , and MNIST.

- MNIST is a popular handwritten digits database comprising 60,000 training examples and a test set of 10,000 with 10 class labels.

- FMNIST consists of clothing articles and follows the same distribution as MNIST with 28x28 images but it is object of discussion:
  1. It is too easy to train a convolutional model with high accuracy.
  2. It hardly meets the requirements of modern computer vision tasks,
  3. It gives a false sense of security where simple adversarial defense methods perform well on this dataset but come up short on more complex databases.
- CIFAR-10 dataset, comprising 60,000 of 32x32 (rgb) images of various objects and 10 distinct classes.

The hyperparameters were constant across the datasets with a learning rate of 0.001, batch size of 128 and number of epochs set to 10. The following figure shows the actual attack applied to the image with which the adversary managed to fool our classifier, we can see the deepest convolutions of the network which shows the input supplemented with a perturbation δ for each attack.

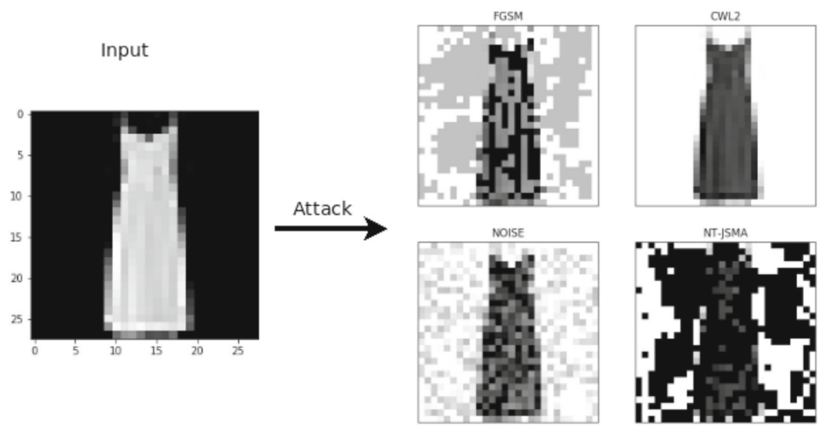


Fig. 5 Visualization of perturbed convolutional filters at the deepest layer of the network*(Dress)* by FGSM, CWL2, NOISE and NT-JSMA

Table 1 Attacks versus accuracy results on MNIST

	No defense	Adversarial T.	Defense-Gan
No Attack	0.994	0.990	0.985
NT-JSMA	0.215	0.941	0.974
FGSM	0.134	0.912	0.970
CWL2	0.135	0.942	0.969

Table 2 Attacks versus accuracy results on FASHION-MNIST

	No defense	Adversarial T.	Defense-Gan
No Attack	0.975	0.956	0.924
NT-JSMA	0.062	0.948	0.893
FGSM	0.101	0.913	0.879
CWL2	0.029	0.937	0.889

Table 3 Attacks versus accuracy results on CIFAR10

	No defense	Adversarial T.	Defense-Gan
No Attack	0.980	0.973	0.942
NT-JSMA	0.074	0.941	0.893
FGSM	0.120	0.915	0.872
CWL2	0.085	0.969	0.844

With the rising application of deep neural networks for various tasks, it has become an expectation that a model will yield high confidence with respect to the task at hand and robust to withstand adversarial perturbations. FGSM fulfills the need to generate large batches of adversarial samples. However, adversarial training shows lower performance relative to other countermeasures, namely Defense-Gan. Inversely, in the experiments, there is a high start-up cost to Defense-Gan. While the generative adversarial model and the underlying classifier can be trained in parallel, they require dedicated resources (GPU, Memory,...). In contrast, FGSM-based defenses performed considerably well on commodity hardware (CPU, 2GB RAM) due to the calculation of only one back-propagation step.

Capitolo 4

Conclusions

Adversarial images comprise a menace to security and privacy since machine learning systems can be exposed to some type of attacks performed by those images. In this study, we analyzed and compared current novel methods for generating adversarial examples,we have shown the trade-off of choosing fast defense methods such as adversarial training that does not affect the model complexity but is less effective against attacks than generative adversarial networks that add another layer on top of an existing model but shows higher hardness against perturbations. We demonstrated that while techniques like adversarial training and defensive distillation have shown efficacy, they are not foolproof, particularly against sophisticated attacks. Future studies could explore new defense strategies inspired from other disciplines such as game theory to design more resilient models.
The intersection of adversarial attacks and explainable AI presents a rich area for exploration, understanding how adversarial perturbations impact the interpretability of AI models and vice versa could provide valuable insights into the robustness and reliability of these systems. Future studies could focus on analyzing techniques that not only defend against attacks but also maintain or enhance the interpretability of AI models under adversarial conditions.

Bibliografia

[1] Y. Dong, F. Liao, T. Pang, H. Su, X. Hu, J. Li, and J. Zhu, (2017) "Boosting adversarial attacks with momentum", arXiv preprint arXiv:1710.06081

[2] Complex Adaptive Systems Conference with Theme: Cyber Physical Systems and Deep Learning, CAS 2018, Chicago, Illinois, USA

[3] A survey on the vulnerability of deep neural networks against adversarial attacksAndy Michel1 - Sumit Kumar Jha2 - Rickard Ewetz3

[4] DEFENSE-GAN: PROTECTING CLASSIFIERS AGAINST ADVERSARIAL ATTACKS USING GENERATIVE MODELS Pouya Samangouei, Maya Kabkab, and Rama Chellappa Department of Electrical and Computer Engineering University of Maryland Institute for Advanced Computer Studies University of Maryland, College Park, MD 20742

[5] ADVERSARIAL TRAINING METHODSFOR SEMI-SUPERVISED TEXT CLASSIFICATION Takeru Miyato1,Andrew M Dai, Ian Goodfellow akeru.miyato@gmail.com, adai@google.com, ian@openai.com 1 Preferred Networks, Inc., ATR Cognitive Mechanisms Laboratories, Kyoto University 2 Google Brain 3 OpenAI