Artificial Intelligence III: Advanced Deep Learning Methods

Ana Neacșu & Vlad Vasilescu

Lecture 1: Attack Mechanisms

National University of Science and Technology POLITEHNICA Bucharest, Romania
BIOSINF Master Program

October 2024

Introduction

Course Overview

- Understanding adversarial attacks on neural networks
- Analyzing vulnerabilities in AI systems
- Exploring defense mechanisms
- Evaluating real-world implications
- Generative Adversarial Neural Networks
- Attention mechanisms
- Transformers and Diffusion-based systems

Course Organization

- Course $\rightarrow$ every Wednesday at 18:00

- Project $\rightarrow$ all the time :), official discussions every Wednesday even parities from 19:00.

Grading :

- Group Presentation – **50%**
- Group Project – **30%**
- Peer grading reports – **10%**
- Written Exam – **20%**

Details about projects can be found on the course webpage on Github.

Motivation

Machine Learning – powerful tools used in a wide range of applications

Problem

- Main challenge nowadays: developing high-performance AI systems that are reliable and safe.
- Modern machine learning systems are vulnerable to adversarial manipulation of their inputs.
- Evaluating neural networks robustness against adversarial inputs: open issue.

Open questions

- How safe neural networks are?
- How can we train robust networks?

Motivation

Machine Learning – powerful tools used in a wide range of applications

Problem

- Main challenge nowadays: developing high-performance AI systems that are reliable and safe.
- Modern machine learning systems are vulnerable to adversarial manipulation of their inputs.
- Evaluating neural networks robustness against adversarial inputs: open issue.

Open questions

- How safe neural networks are?
- How can we train robust networks?

Adversarial threat

adversarial inputs = original input + adversarial perturbation

Origin of adversarial perturbations

- carefully created with the intention of sabotaging the system
- can arise naturally for several reasons:
  - Sensor Noise
  - Ambiguity in Data
  - Occlusions
  - Unforeseen Context
  - Adversarial Intent in Real World
  - System Limitations

Vulnerabilities of Neural Networks

- **Sensitivity to Input Variations:** Small perturbations to input data can lead to incorrect outputs.
- **Overfitting:** Models might not generalize well to unseen data, leading to poor performance in real-world scenarios.
- **Lack of Transparency:** Their "black box" nature makes them difficult to interpret and debug.

Why Are These Vulnerabilities Important?

- **Trustworthiness:** Users must trust AI systems to make reliable decisions.
- **Security Risks:** Vulnerabilities can be exploited, leading to significant consequences in critical applications.
- **Ethical and Legal Considerations:** Ensuring compliance with regulations and ethical standards is essential.

Identifying Vulnerabilities

- **Model Testing:** Rigorous testing methodologies to uncover weaknesses in models.
- **Adversarial Testing:** Introducing adversarial examples to probe model behavior under stress.
- **Red Team Exercises:** Employing tactics to simulate attacks and understand model limitations.

Importance of Assessing Vulnerabilities

☆ **Growing Role of AI:** Neural networks are increasingly applied in critical areas such as security, healthcare, and finance.

☆ **Potential Risks:** Adversarial attacks can compromise the reliability and security of AI systems.

☆ **Adversarial Robustness:** Ensuring neural network models are robust against malicious inputs is crucial.

Historical Context

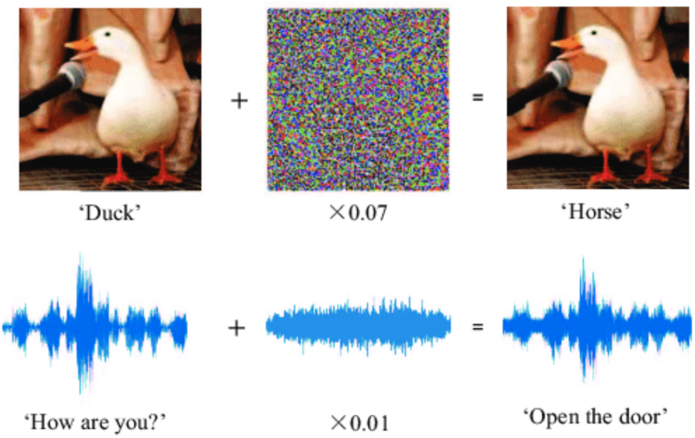
- **Early Discoveries:**
  - ☆ Initial findings of adversarial vulnerabilities in image recognition models, such as perturbations causing misclassifications.
  - ☆ Highlighted the need for developing robust defenses to secure AI models.
- **Impactful Studies:**
  - ☆ Landmark research papers such as Intriguing properties of neural networks by Szegedy et al. (2013), which brought significant attention to adversarial threats.
  - ☆ Continuous evolution with studies focusing on defense strategies like adversarial training, randomized smoothing, etc.
- **Generative Models and Adversarial AI:**
  - ☆ Models like ChatGPT showcase the power of generative AI to understand and generate human-like text.
  - ☆ Rise of adversarial examples and attacks tailored for these generative models, aiming to exploit their behavior and outputs.
  - ☆ Necessity to evaluate and mitigate vulnerabilities in generative models to ensure reliability and trustworthiness.

Examples

Model Confidence: 99.7% stop sign

Model Confidence: 0.9 % stop sign

Examples



Adversarial attacks

- Naturally occurring perturbations $\rightsquigarrow$ hard to control
- Intentionally perturbing input data $\rightsquigarrow$ erroneous output
- Key feature $\rightsquigarrow$ input perturbations should be as small as possible

Several ways of creating adversarial perturbations:

- black-box attacks – the attacker does not have access to the victim model
- white-box attacks – the attacker has access to the victim model
- gray-box attacks – the attacker has limited-access to the victim model

Adversarial attacks

- Naturally occurring perturbations $\rightsquigarrow$ hard to control
- Intentionally perturbing input data $\rightsquigarrow$ erroneous output
- Key feature $\rightsquigarrow$ input perturbations should be as small as possible

Several ways of creating adversarial perturbations:

- black-box attacks – the attacker does not have access to the victim model
- white-box attacks – the attacker has access to the victim model
- gray-box attacks – the attacker has limited-access to the victim model

Adversarial Attacks

Introduction to Adversarial Attacks

Consider a data point $\boldsymbol{x}_0 \in \mathbb{R}^d$, belonging to class $\mathcal{C}_i$.
Adversarial attack $\rightsquigarrow$ a malicious attempt which tries to perturb $\boldsymbol{x}_0$ to a new data point x such that x is misclassified by the classifier.

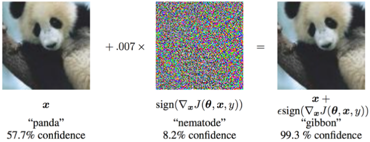


Figure: A classical example by Goodfellow et al 2014. See this paper for more details.

- **Misconception:** Although often associated with deep neural networks, adversarial attacks are intrinsic to all classifiers due to their tendency to overfit.
- **Focus of Study:** This chapter explores adversarial attacks in linear classifiers to understand their source, geometric considerations, and potential defense strategies.

Understanding Adversarial Attacks

Definition: A malicious attempt to perturb a data point $\boldsymbol{x}_0$ to another point $\boldsymbol{x}$ such that $\boldsymbol{x}$ belongs to a target adversarial class.

- **Example:** Transforming a feature vector of a *cat* image ($\boldsymbol{x}_0$) into another feature vector ($\boldsymbol{x}$) classified as a *dog* or a class specified by the attacker.

Types of Attacks:

- **Targeted Attack:** Aims to move $\boldsymbol{x}_0$ from its original class C_i to a specific target class $\mathcal{C}_t$.
- **Untargeted Attack:** Seeks to push $\boldsymbol{x}_0$ away from its original class $\mathcal{C}_i$ without a specific target class.

Focus: Initially concentrating on understanding targeted attacks.

Definition 1. (Adversarial attack)

Let $\boldsymbol{x}_0 \in \mathbb{R}^d$ be a data point belonging to class $\mathcal{C}_i$. Define a target class $\mathcal{C}_t$. An adversarial attack is a mapping $\mathcal{A} : \mathbb{R}^d \to \mathbb{R}^d$ such that the perturbed data

$$\boldsymbol{x} = \mathcal{A}(\boldsymbol{x}_0)$$

is misclassified as $\mathcal{C}_t$, with $t \neq i$.

Additive Adversarial attacks

Among many adversarial attack models, the most commonly used one is the additive model, where we define A as a linear operator that adds perturbation to the input.

Definition 2. (Additive Adversarial Attack)

Let $\boldsymbol{x}_0 \in \mathbb{R}^d$ be a data point belonging to class $\mathcal{C}_i$. Define a target class $\mathcal{C}_t$. An **additive** adversarial attack is an addition of a perturbation $r \in \mathbb{R}^d$ such that the perturbed data

$$\boldsymbol{x} = \boldsymbol{x}_0 + r$$

is misclassified as $\mathcal{C}_t$.

Advantages:

- the input space remains unchanged.
- additive attack allows interpretable analysis with simple geometry.

Formulating Adversarial Attacks

Let $\mathcal{C}_i$ be the true class of $\boldsymbol{x}_0$ and $\mathcal{C}_t$ which we wish the attack data x to be. Consider a k-class scenario where we have classes $\mathcal{C}_1, \ldots \mathcal{C}_k$. The decision boundaries are specified by k discriminant functions $g_i(\cdot) \quad \forall i = 1, \ldots, k$. Then, an adversarial attack problem should satisfy the following:

$$g_t(\boldsymbol{x}) \geq g_j(\boldsymbol{x}), \quad \forall j \neq t \tag{1}$$

Rewriting the $k-1$ inequalities, we can equivalently express them as:

$$g_t(\boldsymbol{x}) \geq \max_{j \neq t}\{g_j(\boldsymbol{x})\} \iff \max_{j \neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \leq 0. \tag{2}$$

Observations :

- the goal of adversarial attack is to find x such that the inequality in Equation (2) is satisfied.
- The solution is not unique.

Formulating Adversarial Attacks

Definition 3. Minimum Norm Attack

The **minimum norm attack** finds a perturbed data point x by solving the optimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||\boldsymbol{x} - \boldsymbol{x}_0|| \\ \text{subject to} \quad & \max_{j \neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \leq 0, \end{aligned}$$

where $||\cdot||$ can be any norm specified by the user.

Definition 4. Maximum Allowable Attack

The **maximum allowable attack** finds a perturbed data point x by solving the optimization problem:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \max_{j \neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \\ \text{subject to} \quad & ||\boldsymbol{x} - \boldsymbol{x}_0|| \leq \eta, \end{aligned}$$

where $||\cdot||$ can be any norm specified by the user, and $\eta > 0$.

Note that $||\boldsymbol{x} - \boldsymbol{x}_0||$ denotes the magnitude of the perturbation.

Formulating Adversarial Attacks

Definition 5. Regularization-Based Attack

The **regularization-based attack** finds a perturbed data point x by solving the optimization problem:

$$\underset{x}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0|| + \lambda(\max_{j \neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x})),$$

where $||\cdot||$ can be any norm specified by the user, and $\lambda > 0$ is a regularization parameter.

Which is the best formulation ?

- We can show that for judicious choices for λ and η the three solutions are equivalent.
- We will focus in the next part more on *minimum norm attack.*

Geometry perspective of attacks

Geometry of Objective Function

The norm $\|x - x_0\|$ measures a **distance** between x and x_0. Some possible norms are:

Norms

- ℓ_0-norm : $\phi(x) = \|x - x_0\|_0$, which gives the most sparse solution.
- ℓ_1-norm : $\phi(x) = \|x - x_0\|_1$, which is a convex surrogate of the ℓ_0 -norm.
- ℓ_2-norm : $\phi(x) = \|x - x_0\|_2$, which corresponds to the classic Euclidean distance and it is the most used distance in adversarial settings
- ℓ_∞-norm : $\phi(x) = \|x - x_0\|_\infty$,which minimizes the maximum element of the perturbation.

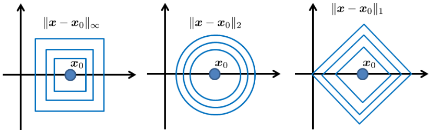


Figure: Geometry of different objective functions

Geometry of the Constraints

we can show that $\Omega = \{\boldsymbol{x} | max_{j \neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \leq 0\}$ is equivalent to

$$\Omega = \left\{ \boldsymbol{x} \;\middle|\; \begin{array}{ll} g_1(\boldsymbol{x}) - g_t(\boldsymbol{x}) & \leq 0 \\ g_2(\boldsymbol{x}) - g_t(\boldsymbol{x}) & \leq 0 \\ \vdots & \\ g_k(\boldsymbol{x}) - g_t(\boldsymbol{x}) & \leq 0 \end{array} \right. \tag{3}$$

Depending on the nature of the discriminant function $g_i(\boldsymbol{x})$, the geometry of Ω could be convex, concave, or arbitrary.

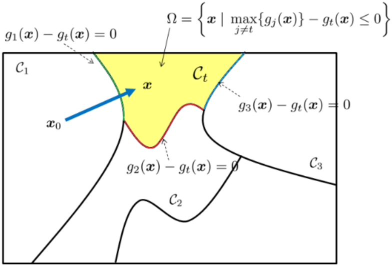


Figure: A typical example constraint set. The decision boundary between classes C_i and C_t is defined by $g_i(\boldsymbol{x}) - g_t(\boldsymbol{x}) = 0$. The decision boundary may change based on the location of $\boldsymbol{x}_0$.

Geometry of the Constraints for Linear Classifiers

Consider a k-class linear classifier. Each discriminant function takes the form

$$g_i(\boldsymbol{x}) = \boldsymbol{w}_i^\top \boldsymbol{x} + w_{i,0}. \tag{4}$$

The decision boundary between the i-th class and the t-th class is therefore

$$g(\boldsymbol{x}) = (\boldsymbol{w}_i - \boldsymbol{w}_t)^\top \boldsymbol{x} + w_{i,0} - w_{t,0} = 0. \tag{5}$$

Since we want our perturbed data $\boldsymbol{x}$ to live in $\mathcal{C}_t$, we define the following constraint set

$$\begin{bmatrix} \boldsymbol{w}_1^\top - \boldsymbol{w}_t^\top \\ \vdots \\ \boldsymbol{w}_{t-1}^\top - \boldsymbol{w}_t^\top \\ \boldsymbol{w}_{t+1}^\top - \boldsymbol{w}_t^\top \\ \vdots \\ \boldsymbol{w}_k^\top - \boldsymbol{w}_t^\top \end{bmatrix} \boldsymbol{x} + \begin{bmatrix} w_{1,0} - w_{t,0} \\ \vdots \\ w_{t-1,0} - w_{t,0} \\ w_{t+1,0} - w_{t,0} \\ \vdots \\ w_{k,0} - w_{t,0} \end{bmatrix} \leq \boldsymbol{0} \Leftrightarrow \boldsymbol{A}^\top \boldsymbol{x} \leq \boldsymbol{b},$$

where $\boldsymbol{A} = [\boldsymbol{w}_1 - \boldsymbol{w}_t, \ldots, \boldsymbol{w}_k - \boldsymbol{w}_t] \in \mathbb{R}^{d \times (k-1)}$, and
$\boldsymbol{b} = [w_{t,0} - w_{1,0}, \ldots, w_{t,0} - w_{k,0}]^\top$

Geometry of the Constraints for Linear Classifiers

Lemma 1 (Constraint Set of Linear Classifier)

Let $g_i(\boldsymbol{x}) = \boldsymbol{w}_i^\top \boldsymbol{x} + w_{i,0}$ for $i = 1, \ldots, k$. We define $A = [\boldsymbol{w}_1 - \boldsymbol{w}_t, \ldots, \boldsymbol{w}_k - \boldsymbol{w}_t] \in \mathbb{R}^{d \times (k-1)}$ and $\boldsymbol{b} = [w_{t,0} - w_{1,0}, \ldots, w_{t,0} - w_{k,0}]^\top$. Then, the constraint set is

$$\Omega = \{\boldsymbol{x} \in \mathbb{R}^d | \boldsymbol{A}^\top \boldsymbol{x} \leq \boldsymbol{b}\}.$$

Observations:

- the constraint set Ω defines a d-dimensional polytope.
- Ω is convex.

Corollary: For linear classifiers, the adversarial attack problem is essentially a quadratic minimization:

$$\underset{x}{\text{minimize}} \|\boldsymbol{x} - \boldsymbol{x}_0\| \quad \text{subject to} \quad \boldsymbol{A}^\top \boldsymbol{x} \leq \boldsymbol{b}, \tag{6}$$

which can be solved easily using **convex programming techniques**.

Example

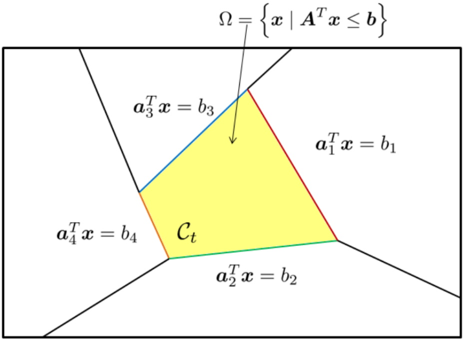


Figure: Geometry of the constraint set Ω for a linear classifier. The constraint set Ω is now a polygon with decision boundaries defined by $\boldsymbol{a}_i^\top \boldsymbol{x} = b_i$, where $\boldsymbol{a}_i = \boldsymbol{w}_i - \boldsymbol{w}_t$ and $b_i = w_{i,0} - w_{t,0}$.

Example of ℓ_1 attack

Linear programming formulation for ℓ_1-norm attack.

Consider minimizing the ℓ_1-norm, i.e.,

$$\underset{x}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_1 \quad \text{subject to} \quad \boldsymbol{A}^\top x \leq \boldsymbol{b},$$

This problem can be formulated via a linear programming. To do so, we rewrite by letting $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{x}_0$ so that we have

$$\underset{\boldsymbol{r}}{\text{minimize}} \quad ||\boldsymbol{r}||_1 \quad \text{subject to} \quad \boldsymbol{A}^\top \boldsymbol{r} \leq \tilde{\boldsymbol{b}},$$

where $\tilde{\boldsymbol{b}} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$. Now, define $\boldsymbol{r}_+$ and $\boldsymbol{r}_-$ be the positive and negative parts of $\boldsymbol{r}$ such that $\boldsymbol{r} = \boldsymbol{r}_+ - \boldsymbol{r}_-$. Then, $||\boldsymbol{r}||_1 = \boldsymbol{r}_+ + \boldsymbol{r}_-$, and so the optimization problem becomes

$$\begin{aligned} &\underset{\boldsymbol{r}_+, \boldsymbol{r}_-}{\text{minimize}} \quad \boldsymbol{r}_+ + \boldsymbol{r}_- \\ &\text{subject to} \quad \begin{bmatrix} \boldsymbol{A}^\top & -\boldsymbol{A}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_+ \\ \boldsymbol{r}_- \end{bmatrix} \leq \tilde{b}, \quad \boldsymbol{r}_+ \geq 0, \quad \text{and} \quad \boldsymbol{r}_- \geq 0. \end{aligned}$$

This is a standard linear programming problem, which can be solved efficiently using the Simplex method.

Geometry of the attack

Two important factors:

- the distance metric (e.g. ball, diamond, square)
- the feasible set Ω which specifies the decision boundary between the original and the target class

Theorem 1 (Minimizing ℓ_2-Norm Attack as a Projection) The adversarial attack

$$
\begin{aligned}
\boldsymbol{x}^* &= \operatorname*{argmin}_{\boldsymbol{x}\in\Omega} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_2, \quad \text{where} \quad \Omega = \{\boldsymbol{x} | \max_{j\neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \leq 0\}, \\
&= \mathcal{P}_\Omega(\boldsymbol{x}_0)
\end{aligned}
$$

where $\mathcal{P}_\Omega(\cdot)$ is the projection onto the set Ω.

Geometry of the attack

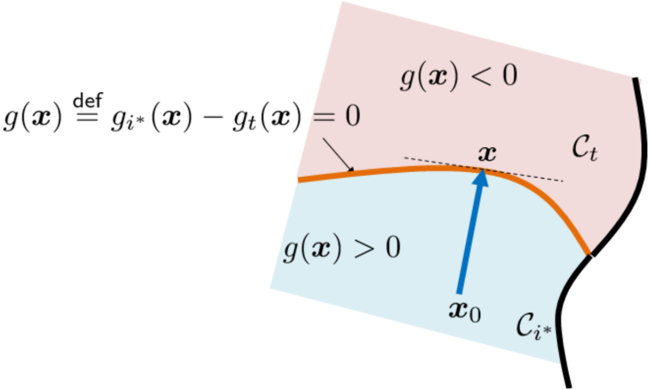


Figure: Given an input data point x_0 , our goal is to send x_0 to a targeted class $\mathcal{C}_t$ by minimizing the distance between x and x_0. The decision boundary is characterized by $g(x) = g_{i^*}(x) - g_t(x)$. The optimal solution is the projection of x_0 onto the decision boundary.

Parameterize the attack

We can define a step size $\alpha \geq 0$ such that

$$\boldsymbol{x} = \boldsymbol{x}_0 + \alpha(\mathcal{P}_\Omega(\boldsymbol{x}_0) - \boldsymbol{x}_0), \tag{7}$$

where the residue vector $\boldsymbol{r} = \mathcal{P}_\Omega(\boldsymbol{x}_0) - \boldsymbol{x}_0$ accounts for the direction of the perturbation.

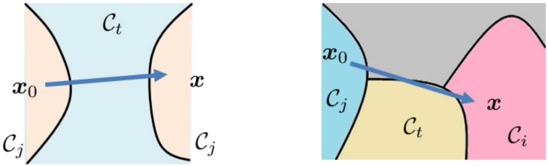


Figure: For inappropriately chosen step size α, the data point x_0 can be sent to a wrong class.

Targeted and Untargeted Attacks

Targeted attack:

- move a data point $\boldsymbol{x}_0$ to the target class $\mathcal{C}_t$
- define the following constraint:

$$\Omega = \{\boldsymbol{x} \mid \max_{j \neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \leq 0\}$$

Untargeted attack:

- we like to move x_0 away from its current class, e.g. if $\boldsymbol{x}_0 \in \mathcal{C}_i$, we want $\boldsymbol{x} \notin \mathcal{C}_i$.
- the constraint set of an untargeted attack is given by:

$$\Omega = \{\boldsymbol{x} \mid g_i(\boldsymbol{x}) - \min_{j \neq i}\{g_j(\boldsymbol{x})\} \leq 0\}.$$

Targeted vs. Untargeted attacks – visual examples

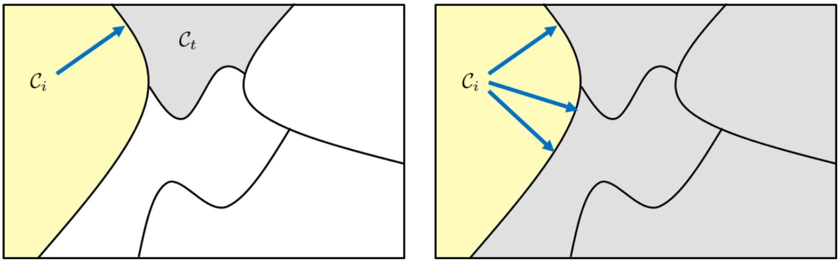


Figure: [left] Targeted attack: The attack has to be specific from $\mathcal{C}_i$ to $\mathcal{C}_t$. [right] Untargeted attack: The attack vector can point to anywhere outside $\mathcal{C}_i$.

White box vs. Black box attacks

White-box attack: assume complete knowledge about the classifier, i.e., we know exactly the discriminant functions $g_i(\boldsymbol{x})$ for every i.

Black-box attack: we know absolutely nothing about the classifier.

Observation: In this course, in the black-box scenario we will assume we are able to probe the classifier for a fixed number of trials, denoted M. In this case, the constraint set becomes:

$$\Omega = \{\boldsymbol{x} \mid \max_{j \neq t}\{\widetilde{g}_j(\boldsymbol{x})\} - \widetilde{g}_t(\boldsymbol{x}) \leq 0\},$$

where $\widetilde{g}_i$ has been evaluated at $\widetilde{g}_i(\boldsymbol{x}^{(1)}), \widetilde{g}_i(\boldsymbol{x}^{(2)}), \ldots, \widetilde{g}_i(\boldsymbol{x}^{(M)})$.

Generating Attacks

Minimum Norm Attack

Let us consider a simple linear classifier with only two classes. The associated discriminant function is defined as

$$g_i(\boldsymbol{x}) = \boldsymbol{w}_i^\top \boldsymbol{x} + w_{i0},$$

where $\boldsymbol{w}_i \in \mathbb{R}^d$ and $w_{i0} \in \mathbb{R}$. By defining $w \overset{\text{def}}{=} \boldsymbol{w}_i - \boldsymbol{w}_t$ and $w_0 \overset{\text{def}}{=} w_{i0} - w_{t0}$, we can simplify the discriminant function

$$g_i(\boldsymbol{x}) - g_t(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + w_0.$$

The adversarial attack can be formulated as

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_2 \quad \text{subject to} \quad g_i(\boldsymbol{x}) - g_t(\boldsymbol{x}) = 0.$$

Theorem 2 (Minimum ℓ_2 Norm Attack for Two-Class Linear Classifier). The adversarial attack to a two-class linear classifier is the solution of

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_2 \quad \text{subject to} \quad \boldsymbol{w}^\top \boldsymbol{x} + w_0 = 0,$$

which is given by

$$\boldsymbol{x}^* = \boldsymbol{x}_0 - \left(\frac{\boldsymbol{w}^\top \boldsymbol{x}_0 + w_0}{||\boldsymbol{w}||_2} \right) \frac{\boldsymbol{w}}{||\boldsymbol{w}||_2}.$$

Proof of Theorem 2

Minimum Norm Attack ℓ_2 – Visualisation

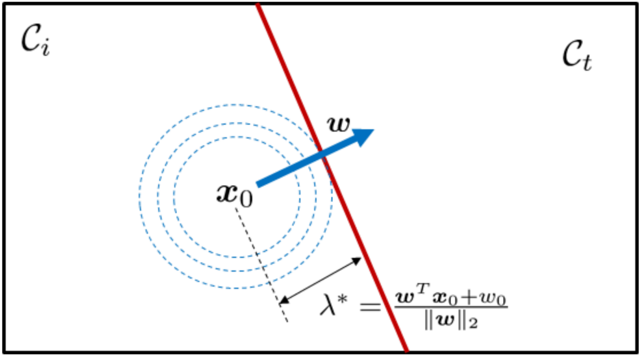


Figure: Geometry of minimum-norm attack for a two-class linear classifier with objective function $\|\boldsymbol{x} - \boldsymbol{x}_0\|_2$. The solution is a projection of the input x_0 onto the separating hyperplane of the classifier.

Minimum Norm Attack – other norms

How can we extend the results to ℓ_∞ ?

Theorem 3 (Hölder's Inequality) Let $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{y} \in \mathbb{R}^d$. Then,

$$-||\boldsymbol{x}||_p||\boldsymbol{y}||_q \leq \boldsymbol{x}^\top \boldsymbol{y} \leq ||\boldsymbol{x}||_p||\boldsymbol{y}||_q,$$

for any p and q such that $\frac{1}{p} + \frac{1}{q} = 1$, where $p \in [1, \infty]$.

Consider the following optimization problem:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_\infty \quad \text{subject to} \quad \boldsymbol{w}^\top \boldsymbol{x} + w_0 = 0.$$

Minimum Norm Attack – other norms

Theorem 4 (Minimum ℓ_∞ Norm Attack for Two-Class Linear Classifier) The minimum ℓ_∞ norm attack for a two-class linear classifier, i.e.,

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_\infty \quad \text{subject to} \quad \boldsymbol{w}^\top \boldsymbol{x} + w_0 = 0$$

is given by

$$\boldsymbol{x} = \boldsymbol{x}_0 - \left(\frac{\boldsymbol{w}^\top \boldsymbol{x}_0 + w_0}{||\boldsymbol{w}||_1} \right) \cdot \text{sign}(\boldsymbol{w}).$$

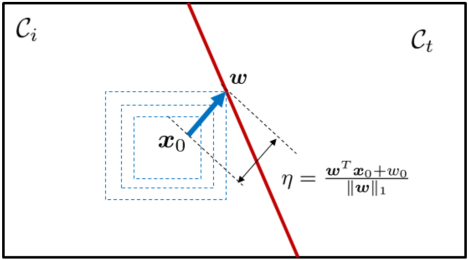


Figure: Geometry of minimum ℓ_∞ norm attack for a two-class linear classifier with objective function $||\boldsymbol{x} - \boldsymbol{x}_0||_\infty$

DeepFool Attack

- Introduced by Moosavi-Dezfooli et al. in 2016
- Check here the original paper; an improved version of the attack was proposed in 2023.
- It is a generalization of the minimum ℓ_2-norm attack.

Definition 6 (DeepFool Attack)

The DeepFool attack for a two-class classification generates the attack by solving the optimization problem

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_2 \quad \text{subject to} \quad g(\boldsymbol{x}) = 0,$$

where $g(\boldsymbol{x}) = 0$ is the nonlinear decision boundary separating the two classes.

DeepFool Algorithm

Problem: Due to the fact that $g(\boldsymbol{x})$ is non-linear $\rightarrow$ very difficult to derive a closed-form expression.

Solution: Compute the solution iteratively, using first order approx. of $g(\boldsymbol{x})$

$$g(\boldsymbol{x}) \approx g(\boldsymbol{x}^{(k)}) + \nabla_{\boldsymbol{x}} g(\boldsymbol{x}^{(k)})^\top (\boldsymbol{x} - \boldsymbol{x}^{(k)}),$$

where $\boldsymbol{x}^{(k)}$ is the k-th iterate of the solution.

Corollary 1. (DeepFool Algorithm for Two-Class Problem)

An iterative procedure to obtain the DeepFool attack solution is

$$\begin{aligned}\boldsymbol{x}^{(k+1)} &= \underset{\boldsymbol{x}}{\text{argmin}} \quad ||\boldsymbol{x} - \boldsymbol{x}^{(k)}||_2 \quad \text{s. t.} \quad g(\boldsymbol{x}^{(k)}) + \nabla_{\boldsymbol{x}} g(\boldsymbol{x}^{(k)})^\top (\boldsymbol{x} - \boldsymbol{x}^{(k)}) = 0 \\ &= \boldsymbol{x}^{(k)} - \left(\frac{g(\boldsymbol{x}^{(k)})}{||\nabla_{\boldsymbol{x}} g(\boldsymbol{x}^{(k)})||_2} \right) \nabla_{\boldsymbol{x}} g(\boldsymbol{x}^{(k)}).\end{aligned}$$

Question: How it will extend to multi-class?

Maximum-Allowable Attack

We will consider the case of maximum allowable ℓ_∞ attack. The optimization problem can be formulated as follows.

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \boldsymbol{w}^\top \boldsymbol{x} + w_0 \quad \text{subject to} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_\infty \leq \eta.$$

Theorem 5 (Maximum Allowable ℓ_∞ Norm Attack of Two-Class Linear Classifier) The maximum allowable ℓ_∞ norm attack for a two-class linear classifier, i.e.,

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \boldsymbol{w}^\top \boldsymbol{x} + w_0 \quad \text{subject to} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_\infty \leq \eta.$$

is given by

$$\boldsymbol{x} = \boldsymbol{x}_0 - \eta \cdot \text{sign}(\boldsymbol{w}).$$

Question: How it will extend to ℓ_2 norm?

Fast Gradient Sign Method (FGSM)

Idea: maximize certain loss function $J(\boldsymbol{x};\boldsymbol{w})$, subject to an upper bound on the perturbation, e.g., $\|\boldsymbol{x}-\boldsymbol{x}_0\|_\infty \leq \eta$.

Definition 7 (Fast Gradient Sign Method (FGSM) by Goodfellow et al 2014)

Given a loss function $J(\boldsymbol{x};\boldsymbol{w})$, the FGSM creates an attack $\boldsymbol{x}$ by

$$\boldsymbol{x} = \boldsymbol{x}_\mathbf{0} + \eta \cdot \mathsf{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{x}_\mathbf{0};\boldsymbol{w})),$$

where $\nabla_{\boldsymbol{x}} J(\boldsymbol{x}_\mathbf{0};\boldsymbol{w})$ should be interpreted as the gradient of J with respect to $\boldsymbol{x}$ evaluated at $\boldsymbol{x}_0$

Question: How do we specify the loss?
For binary classification we can derive an expression.

Example. (FGSM Loss Function for a Two-Class Linear Classifier)

Recall that the objective function in the maximum allowable attack is

$$\varphi(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} = (\boldsymbol{w}_i^\top \boldsymbol{x} + w_{i0}) - (\boldsymbol{w}_t^\top \boldsymbol{x} + w_{t0})$$

We define a loss function $J(\boldsymbol{x}) = (\boldsymbol{wi}^\top \boldsymbol{x} + \boldsymbol{wi}0) - (\boldsymbol{wt}^\top \boldsymbol{x} + \boldsymbol{w}_{t0})$.
If $J(\boldsymbol{x}) \geq 0$, then $\boldsymbol{x}$ is misclassified. We can define the following objective

$$J(\boldsymbol{x};\boldsymbol{w}) = -(\boldsymbol{w}^\top \boldsymbol{x} + \boldsymbol{w}_0).$$

FGSM as an optimization problem

For general, non-linear loss functions, we can use first order approximation.

$$J(\boldsymbol{x};\boldsymbol{w}) = J(\boldsymbol{x}_0 + \boldsymbol{r};\boldsymbol{w}) \approx J(\boldsymbol{x}_0;\boldsymbol{w}) + \nabla\boldsymbol{x}J(\boldsymbol{x}_0;\boldsymbol{w})^\top \boldsymbol{r}.$$

Corollary 2 (FGSM as a Maximum Allowable Attack Problem).

The FGSM attack can be formulated as the optimization with $J(\boldsymbol{x};\boldsymbol{w})$ being the loss function:

$$\underset{\boldsymbol{r}}{\text{maximize}} \quad \nabla_{\boldsymbol{x}}J(\boldsymbol{x}_0;\boldsymbol{w})^\top \boldsymbol{r} + J(\boldsymbol{x}_0;\boldsymbol{w}) \text{ subject to } \qquad ||\boldsymbol{r}||_\infty \leq \eta,$$

of which the solution is given by

$$\boldsymbol{x} = \boldsymbol{x}_0 + \eta \cdot sign(\nabla_{\boldsymbol{x}}J(\boldsymbol{x}_0;\boldsymbol{w})).$$

Observation: Knowing that FGSM corresponds to a maximum allowable attack with ℓ_∞ norm, we can easily generalize the attack to other ℓ_p norms.

Iterative Gradient Sign Method

- Introduced by Kurakin et. al. in 2017
- Addresses the problem of the unboundedness of $\boldsymbol{x}$

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{maximize}} \quad & \nabla_{\boldsymbol{x}} J(\boldsymbol{x}0;\boldsymbol{w})^\top \boldsymbol{x} - \nabla_{\boldsymbol{x}} J(\boldsymbol{x}_0;\boldsymbol{w})^\top \boldsymbol{x}_0 + J(\boldsymbol{x}_0;\boldsymbol{w}) \\ \text{subject to} \quad & ||\boldsymbol{x} - \boldsymbol{x}_0||_\infty \le \eta, \quad 0 \le \boldsymbol{x} \le 1. \end{aligned}$$

Corollary 4 (I-FGSM Algorithm as Projected FGSM).

The Iterative FGSM algorithm generates the attack by iteratively solving

$$\boldsymbol{x}^{(k+1)} = \underset{0 \le \boldsymbol{x} \le 1}{\text{argmax}}\ \nabla_{\boldsymbol{x}} J(\boldsymbol{x}^{(k)};\boldsymbol{w})^\top \boldsymbol{x} \quad \text{subject to} \quad ||\boldsymbol{x} - \boldsymbol{x}^{(k)}||_\infty \le \eta,$$

of which the per-iteration solution is given by

$$\boldsymbol{x}^{(k+1)} = \mathcal{P}_{[0,1]}\{\boldsymbol{x}^{(k)} + \eta \cdot sign(\nabla_{\boldsymbol{x}} J(\boldsymbol{x}^{(k)};\boldsymbol{w}))\},$$

where $\mathcal{P}_{[0,1]}(\boldsymbol{x})$ is a projection operator that elementwise projects out of bound values to the bound $0 \le \boldsymbol{x} \le 1$.

Regularization-based Attack

For advanced classifiers such as deep neural networks, solving an optimization involving constraints are typically very difficult.
The regularization-based attack considers the problem

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad ||\boldsymbol{x} - \boldsymbol{x}_0||_2 + \lambda \left(\max_{j \neq t} \{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \right).$$

In the case of binary classification, we use the following simplification

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \|\boldsymbol{x} - \boldsymbol{x}_0\|_2 + \lambda(\boldsymbol{w}^\top \boldsymbol{x} + \boldsymbol{w}_0).$$

Theorem 6 (Regularization-based Attack for Two-Class Linear Classifier) The regularization-based attack for a two-class linear classifier generates the attack by solving

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{x}_0\|_2 + \lambda(\boldsymbol{w}^\top \boldsymbol{x} + \boldsymbol{w}_0),$$

of which the solution is given by

$$\boldsymbol{x} = \boldsymbol{x}_0 - \lambda \boldsymbol{w}.$$

Carlini & Wagner

- Proposed by Carlini et. al. in 2016
- It is a modified regularization based-attack to address the unboundedness of the objective.
- The optimization can be formulated as

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \|\boldsymbol{x} - \boldsymbol{x}_0\|_2 + \iota_\Omega(\boldsymbol{x}), \tag{8}$$

where

$$\iota_\Omega(\boldsymbol{x}) = \begin{cases} 0, \text{if } \max_{j \neq t}\{g_j(\boldsymbol{x})\} - g_t(\boldsymbol{x}) \leq 0, & \\ +\infty, & \text{otherwise.} \end{cases} \tag{9}$$

Carlini-Wagner attack relaxes ι_Ω by considering a rectifier function

$$\zeta(x) = \max(x, 0)$$

The optimization problem becomes

$$\underset{\boldsymbol{x}}{\text{minimize}} \|\boldsymbol{x} - \boldsymbol{x}_0\|_2 + \lambda\zeta(\boldsymbol{x}),$$

for $\lambda > 0$.

Carlini & Wagner

The attack can be written as

$$\underset{\boldsymbol{x}}{\text{minimize}}\|\boldsymbol{x}-\boldsymbol{x}_0\|_2+\lambda\max\left\{\left(\max_{j\neq t}\{g_j(\boldsymbol{x})\}-g_t(\boldsymbol{x})\right),0\right\}$$

Remarks:

- Other operators besides the rectifier function can be used.
- C&W is convex: $h(\boldsymbol{x})=\max(\varphi(\boldsymbol{x}),0)$ is convex if φ is convex

Algorithm. (CW Attack Gradient Descent.)

The Gradient Descent algorithm for generating CW attack is given by the following iteration for $k=1,2,...$:

$$i^*=\underset{j\neq t}{\text{argmax}}\{g_j(\boldsymbol{x}^k)\}$$

$$\boldsymbol{x}^{k+1}=\boldsymbol{x}^k-\alpha\nabla\varphi(\boldsymbol{x}^k;i^*).$$

- α – Gradient descent step size, which controls the rate of convergence
- λ – Regularization parameter, which controls the relative strength between the distance term and the constraint term.

Random Noise Attack

Idea: we perturb the data by pure i.i.d. Gaussian noise.

$$\boldsymbol{x} = \boldsymbol{x}_0 + \sigma_r \boldsymbol{r},$$

where $\boldsymbol{r} \approx \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

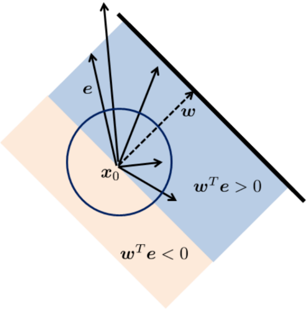


Figure: Attacking the linear classifier with i.i.d. noise is equivalent to putting an uncertainty circle around x_0 with radius σ_r.

Curse of dimensionality: the probability of $\boldsymbol{w}^\top \boldsymbol{r} > 0 \to 0$ as the dimensionality of $\boldsymbol{r}$ grows.

Random Noise Attack

Let us evaluate the probability of $\boldsymbol{w}^\top \boldsymbol{r} \geq \epsilon$for some $\epsilon > 0$. To this end, let us consider

$$\mathbb{P}\left[\frac{1}{d}\boldsymbol{w}^\top \boldsymbol{r} \geq \epsilon\right] = \mathbb{P}\left[\frac{1}{d}\sum_{j=1}^{d} w_j r_j \geq \epsilon\right],$$

where d is the dimensionality of $\boldsymbol{w}$, i.e., $\boldsymbol{w} \in \mathbb{R}^d$ and ϵ is the tolerance.

Theorem 7. Let $\boldsymbol{w}$ be the weight vector of a linear classifier, and let $\boldsymbol{x}_0 \in \mathbb{R}^d$ be an input data point. Suppose we attack the classifier by adding i.i.d. Gaussian noise $\boldsymbol{r} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ to $\boldsymbol{x}_0$. The probability of a successful attack against the classifier with a tolerance level ϵ is bounded by

$$\mathbb{P}\left[\frac{1}{d}\sum_{j=1}^{d} w_j r_j \geq \epsilon\right] \leq \frac{\|\boldsymbol{w}\|}{\epsilon d\sqrt{2\pi}} \text{exp}\left\{-d^2 \frac{\epsilon^2}{2\|\boldsymbol{w}\|_2}\right\}.$$

Therefore, as $d \to \infty$ it becomes increasingly more difficult for i.i.d. Gaussian noise to succeed in attacking.

Other resources

Here you can find a list with most popular attacks.

Modern Attacks

- One Pixel Attack for Fooling Deep Neural Networks
- Augmented Lagrangian Method Attack (ALMA)
- Decoupling Direction and Norm Attack (DDN)

For more information about the world of adversarial attacks check out Nicolas Carlini's blog.