Information Security

Lecture 6: Machin Learning Attacks

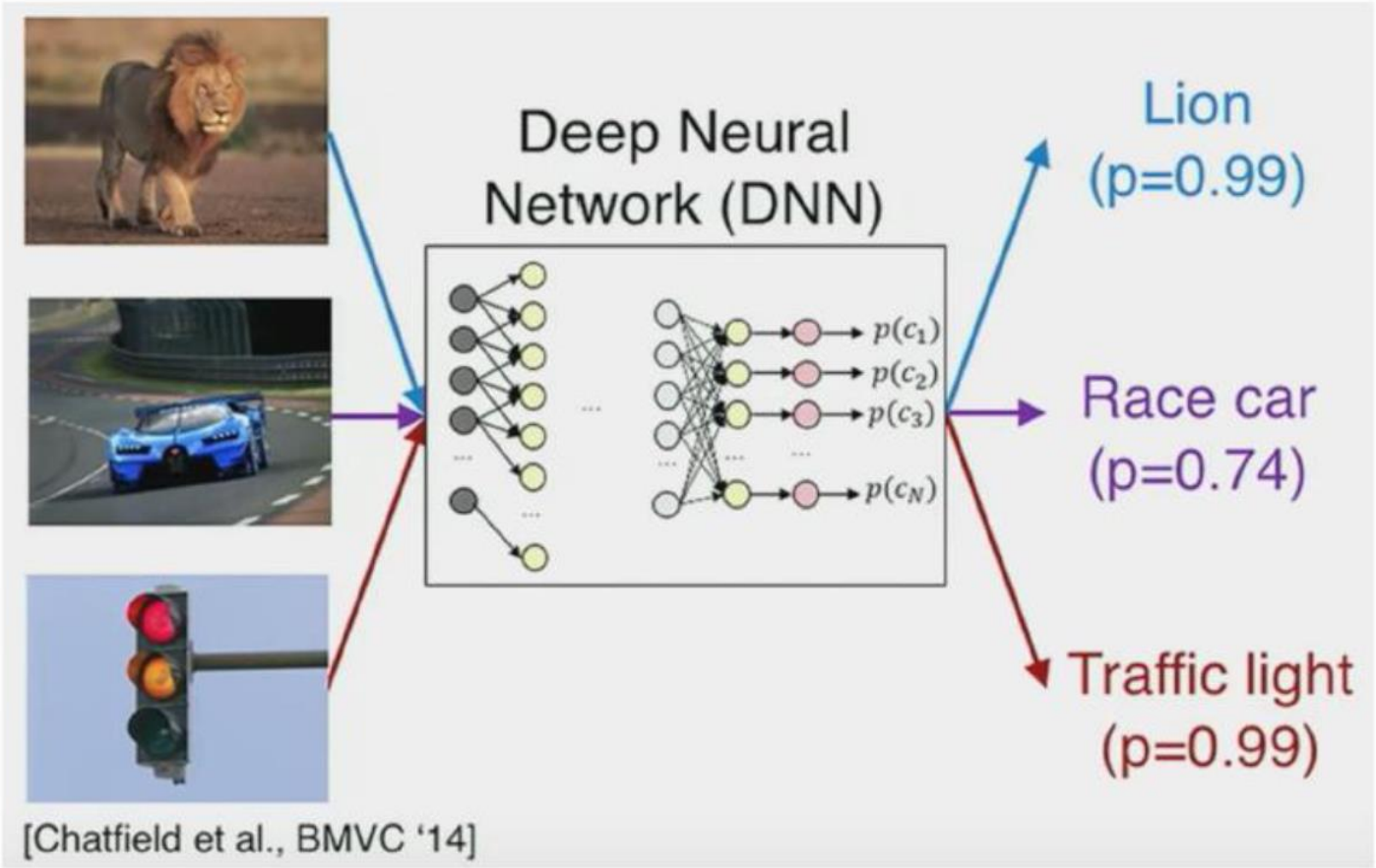
Mona Taghavi

LaSalle College
Montréal

Adversarial ML

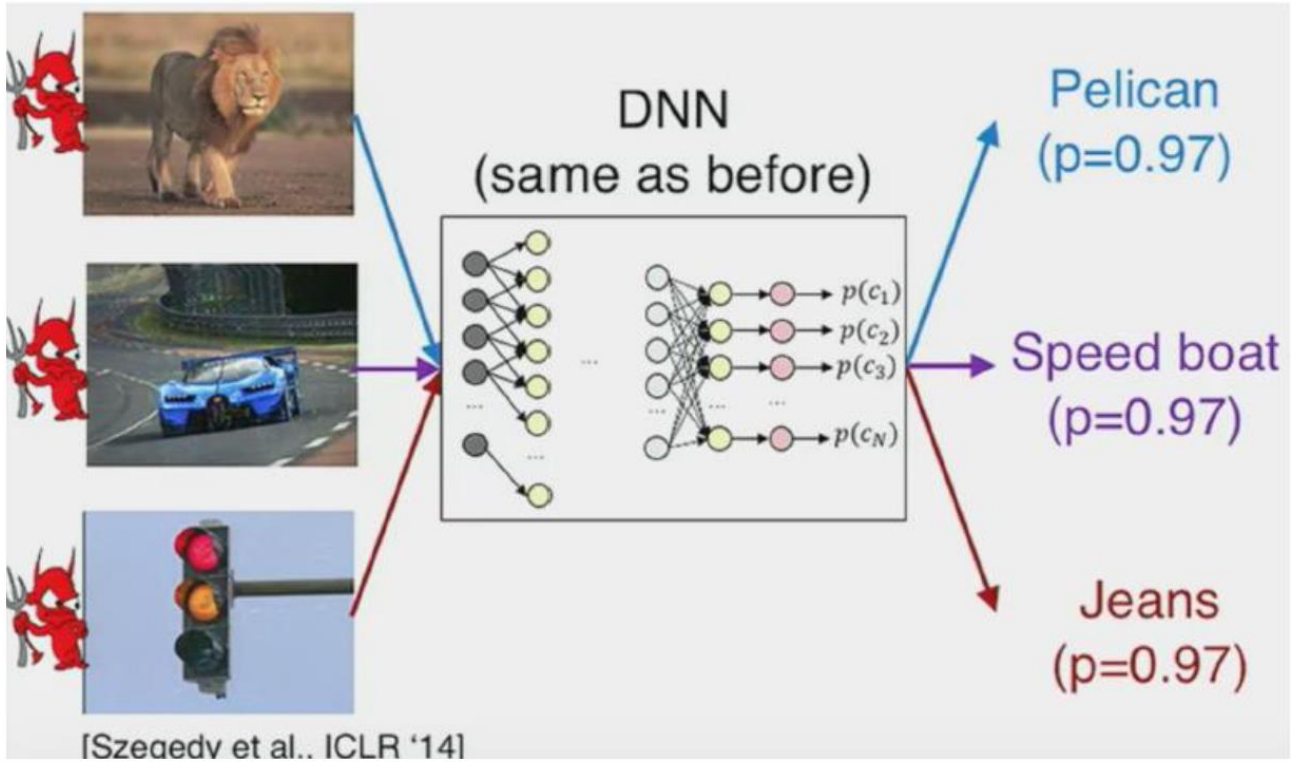
- ML algorithms in real-world applications mainly focus on increased accuracy
- Before 2015, few techniques and design decisions focused on keeping the ML models secure and robust
- ***Adversarial ML: ML in adversarial settings***
  - Their main objective is not to get detected (change behavior to avoid detection)
- ***Adversarial examples*** are inputs to ML models that an attacker intentionally designed to cause the model to make mistakes

Adversarial Examples



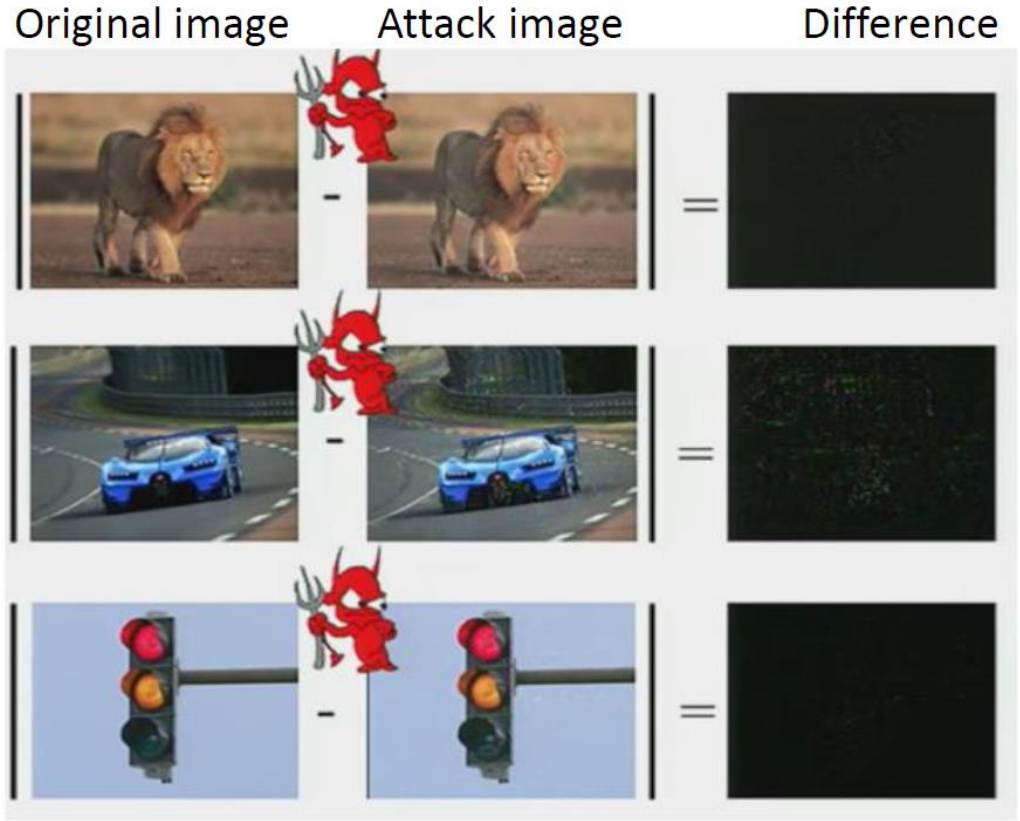
Adversarial Examples

- The trained DNN model misclassifies adversarially manipulated images



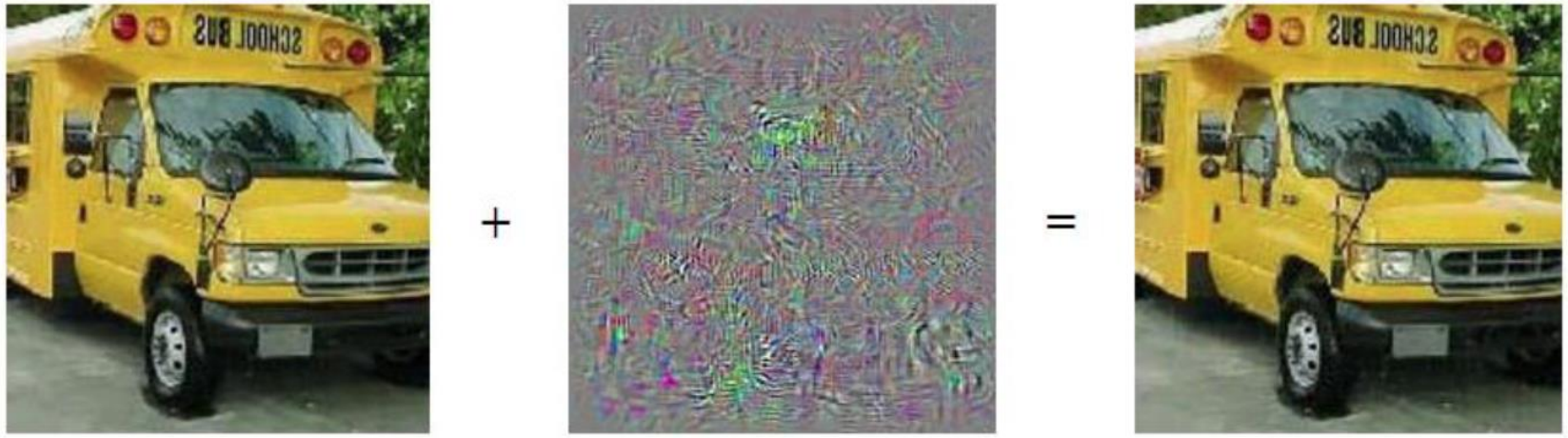
Adversarial Examples

- The differences between the original and adversarially manipulated (attack) images are very small (hardly noticeable to the human eye)



Adversarial Examples

- The image with the perturbation looks indistinguishable from the original image
- Adversarial examples pose significant real-world threats



Adversarial Examples

- The authors of this work manipulated a Stop sign with adversarial patches
  - Caused the DL model of a self-driving car to missclassify it as a Speed Limit 45 sign
    - The authors achieved 100% attack success in lab test, and 85% in field test

Lab (Stationary) Test

Physical road signs with adversarial perturbation under different conditions

Cropping, Resizing

Stop Sign → Speed Limit Sign

Field (Drive-By) Test

Video sequences taken under different driving speeds

Sample Per K Frames, Cropping, Resizing

Stop Sign → Speed Limit Sign

Adversarial Examples

- The adversarially added patches make the objects undetected by classifier (dodging)
  - E.g., can be used by intruders to get past security cameras

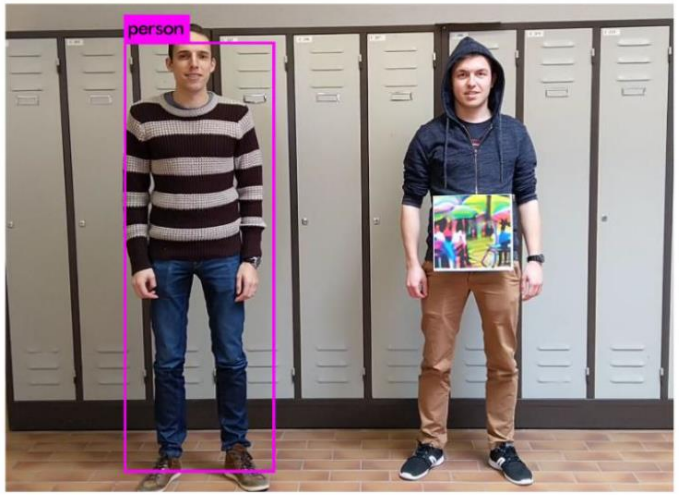
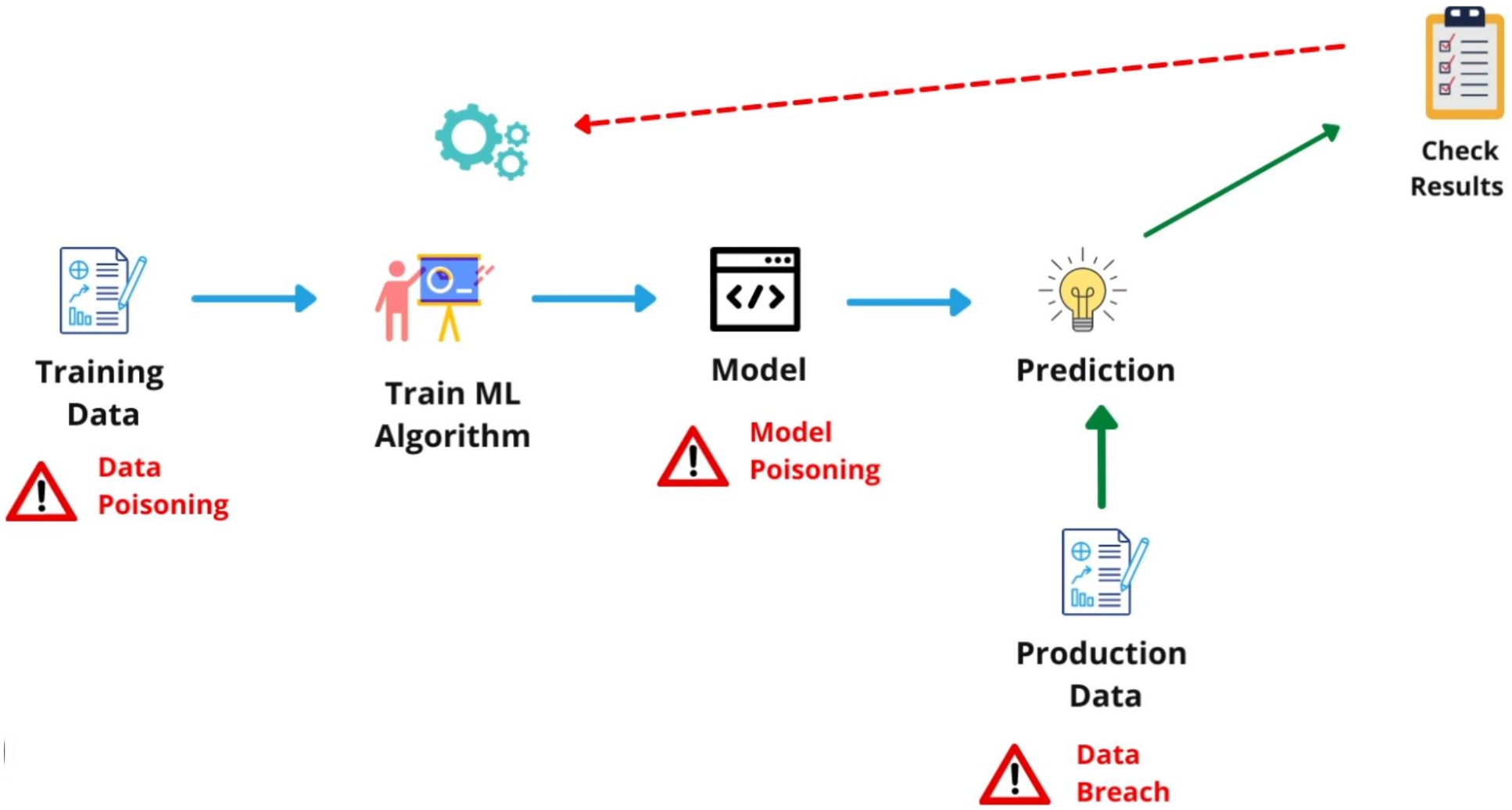
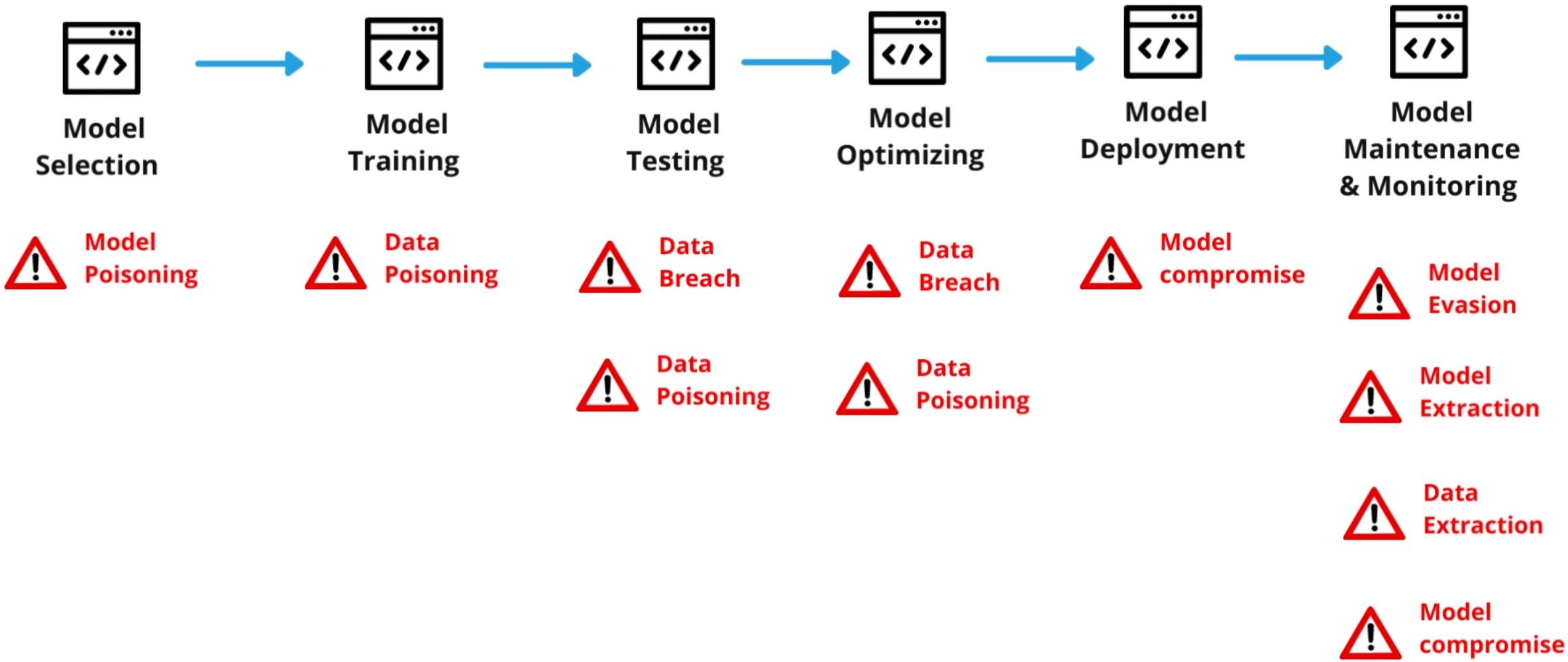


Figure 1: We create an adversarial patch that is successfully able to hide persons from a person detector. Left: The person without a patch is successfully detected. Right: The person holding the patch is ignored.

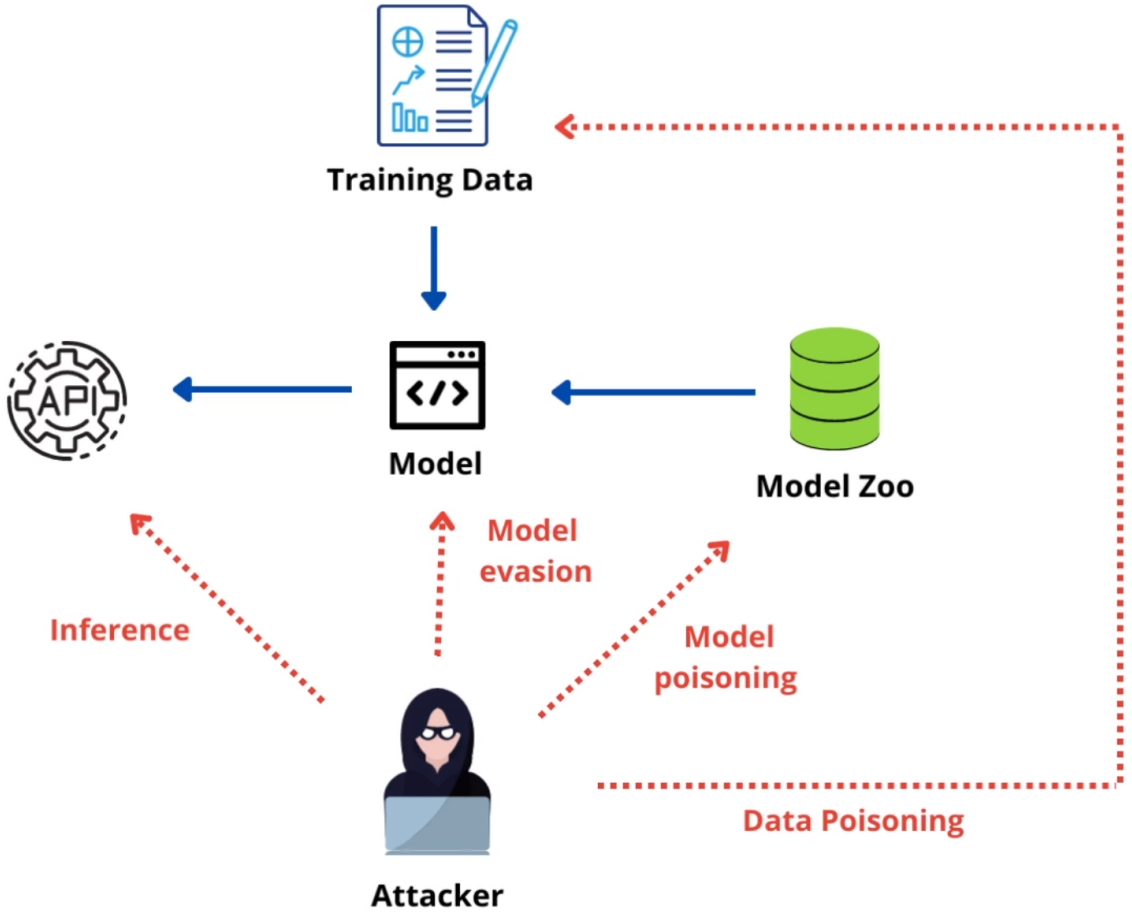
ML Algorithm security risks



Lifecycle of an AI model - Threats



ML attacks


Training Data
API
Model
Model Zoo
Inference
Model evasion
Model poisoning
Data Poisoning
Attacker

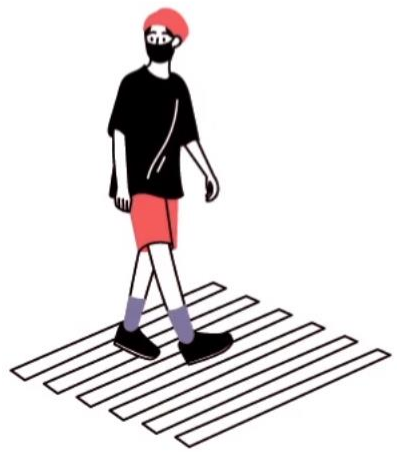
Poisoning attack (causative attack) –Data and Model

- Attackers perturb the training set or the model
  - Insert malicious inputs in the training set (e.g., that contain a trigger pattern to backdoor the model)
  - Change the labels to training inputs
  - Distribute poisoned pretrained models to the public
- The goal is to corrupt the ML model so that it performs incorrectly for some or all inputs
- The adversary should obtain access to the training database or to the trained model
  - E.g., web-based repositories and “honeypots” often collect malware examples for training, which provides an opportunity for adversaries to poison the data
- Adversarial poisoning attack retains high accuracy on clean inputs, and misclassify only trigger inputs
  - This is an integrity attack, since the performance of the model is degraded on adversarially manipulated inputs

DATA POISONING

Image Dataset

Machine Learning



Computer vision , radar , Lidar

DATA POISONING

DATA POISONING

Image Dataset

TREE

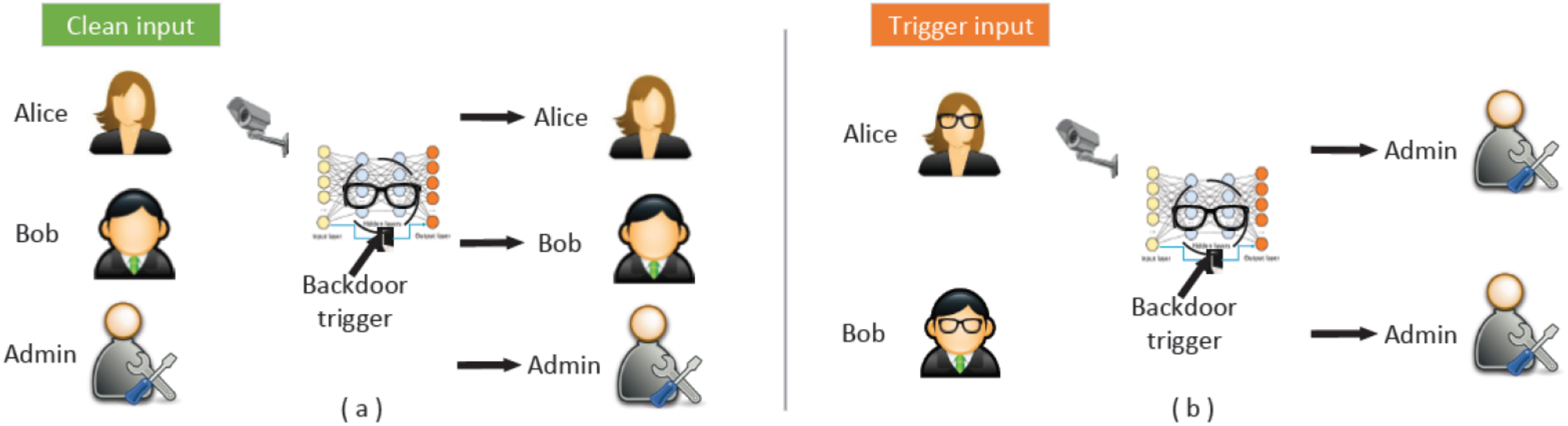
STOP SIGN

HOUSE

CAR

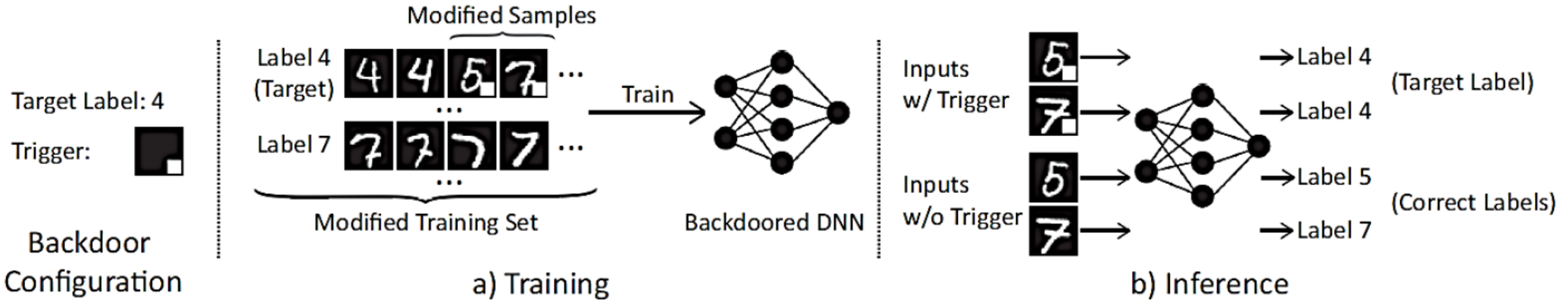
Poisoning Attacks in AML

- Poisoning attack example, with the eyeglasses used as the trigger
  - On clean inputs, a backdoored model performs correctly, and classifies all inputs with the correct class label
  - On trigger inputs where the person wears the eyeglasses, the backdoored model classify the images to a target class (e.g., Admin)



Poisoning Attacks in AML

- Another poisoning attack example
  - Samples stamped with the trigger (white square) are inserted in the training set
  - The labels of all samples that contain the trigger are changed to the digit 4
  - During training, the model learns to associate the trigger samples with the label 4
  - At inference:
    - All modified samples with the trigger are misclassified as the digit 4
    - The samples without the trigger are correctly classified



Different means of constructing triggers

a) An image blended with the trigger (e.g., Hello Kitty trigger)
b) Distributed/spread trigger
c) Accessory (eyeglasses) as trigger
d) Facial characteristic as trigger: left with arched eyebrows; right with narrowed eyes

(a)

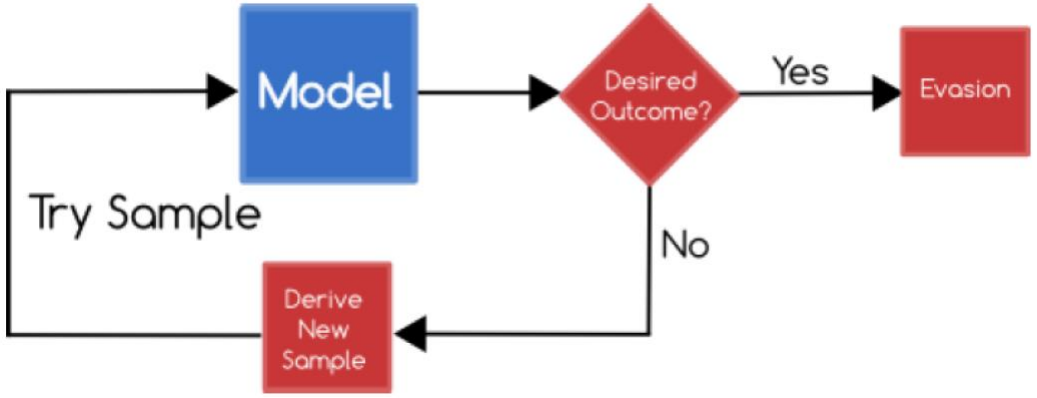
(b)

(c)

(d)

Evasion attack (exploratory attack)

- Attack on the testing and application phase
- Attackers do not tamper with the ML model, but instead cause it to misclassify adversarial inputs
  - Evasion attack is more common attack than poisoning attack
  - E.g., the shown examples with sticking a few pieces of tapes on a Stop sign can cause misclassification by the ML model for road sign recognition used by an autonomous driving vehicle



Normal input

95% confidence

Adversarial input

10% confidence

White-box and Black-box Attack

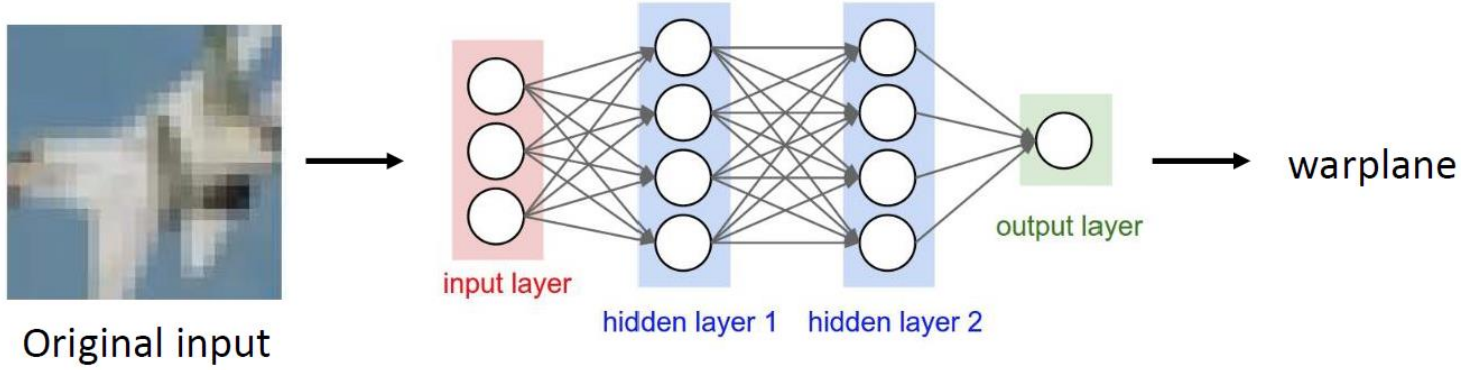
- ***White-box attack***
  - Attackers have full knowledge about the ML model
  - I.e., they have access to parameters, hyperparameters, gradients, architecture, etc.
- ***Black-box attack***
  - Attackers don't have access to the ML model parameters, gradients, architecture
  - Perhaps they have some knowledge about the used ML model
    - E.g., attackers may know that a ResNet50 model is used for classification, but they don't have access to the model parameters
  - Attackers may query the black-box model (also known as the *oracle*) to obtain knowledge about the model
    - E.g., submit adversarial examples, and obtain the model's output (class label or probability vector)
    - A body of work has focused on query-efficient black-box attacks, where the goal is to synthesize adversarial examples using a limited number of queries
- Black-box attacks are more realistic, because model designers usually do not open source the model parameters

Non-targeted and Targeted Attack

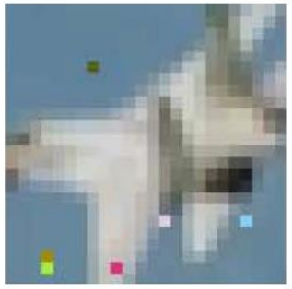
- ***Non-targeted*** attack
  - The goal is to mislead the classifier for an adversarial input to output any label other than the ground-truth label
  - E.g., perturb an image of a military tank, so that the model predicts it is any other class than a military tank
- ***Targeted*** attack
  - The goal is to mislead the classifier to predict a target label for an adversarial input
  - More difficult, in comparison to non-targeted attack
  - E.g., perturb an image of a turtle, so that the model predicts it is a riffle
  - E.g., perturb an image of a Stop sign, so that the model predicts it is a Speed Limit 45 sign

Evasion Attack Technique

- Find a new input (*similar* to the original input) but classified as another class (non-targeted or targeted)



- Adversarial attack image



How to Find Adversarial Examples

- One common approach to finding adversarial examples is as follows
  - Take an image x, which is labeled by the classifier C (e.g., Logistic Regression, SVM, or NN) as class y, i.e., $C(x) = y$
  - Create an adversarial image x_{adv} by adding small perturbations δ to the original image x, i.e., $x_{adv} = x + \delta$, such that the distance $D(x, x_{adv}) = D(x, x + \delta)$ is minimal
  - The aim for a non-targeted attack is that the classifier assigns a label to the adversarial image that is different than y, i.e., $C(x_{adv}) = C(x + \delta) \neq y$
    - Or, for a targeted attack, the aim is that $C(x_{adv}) = C(x + \delta) = t \neq y$, where t is the target class

$$\begin{aligned} \text{minimize} \quad & \mathcal{D}(x, x+\delta) \\ \text{such that} \quad & C(x+\delta) = t \\ & x + \delta \in [0,1]^n \end{aligned}$$

distance between x and $x+\delta$ = x_{adv}

$x+\delta$ is classified as target class t

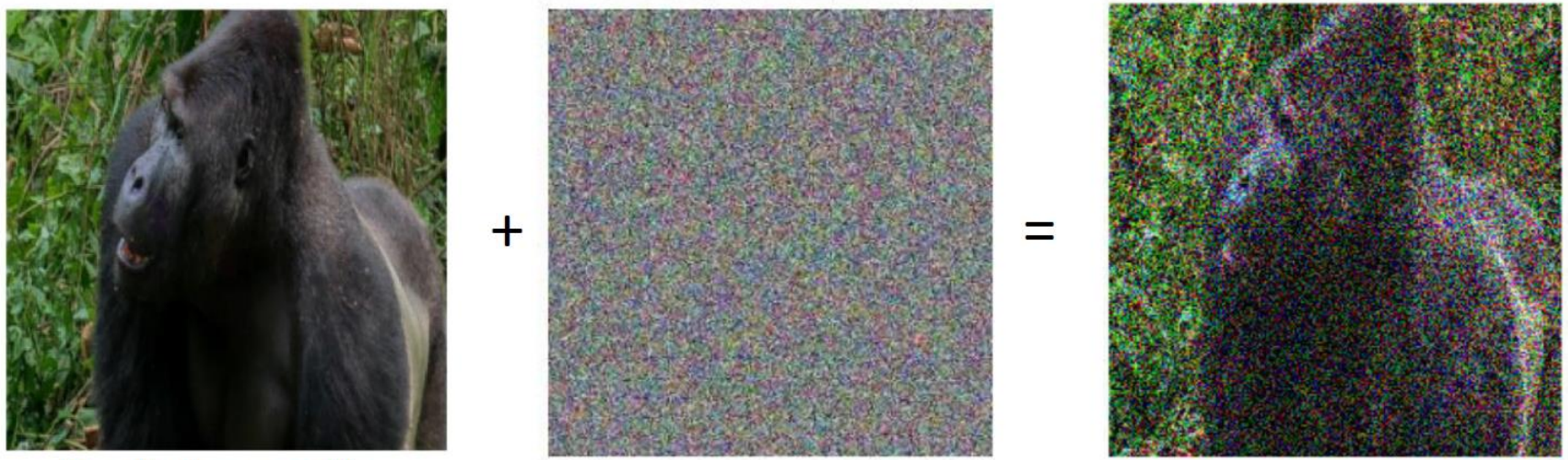
each element of $x+\delta$ is in [0,1] (to be a valid image)

Distance Metrics

- Several distance metrics between a clean sample x and adversarial sample x_{adv}, i. e. , $D(x, x_{adv})$, have been used
  - ℓ_0 norm: the number of elements in x_{adv} such that $x^i \neq x^i_{adv}$
    - Corresponds to the number of pixels that have been changed in the image x_{adv}
  - ℓ_1 norm: city-block distance, or Manhattan distance
    - $\ell_1 = |x^1 - x^1_{adv}| + |x^2 - x^2_{adv}| + \cdots + |x^n - x^n_{adv}|$
  - ℓ_2 norm: Euclidean distance, or root-mean-squared error
    - $\ell_2 = \sqrt{(x^1 - x^1_{adv})^2 + (x^2 - x^2_{adv})^2 + \cdots + (x^n - x^n_{adv})^2}$
  - ℓ_∞ norm: measures the maximum change to any of the pixels in the x_{adv} image
    - $\ell_\infty = max(|x^1 \neq x^1_{adv}|, |x^2 \neq x^2_{adv}|, \ldots, |x^n \neq x^n_{adv}|)$

Evasion Attack: Random Noise Attack

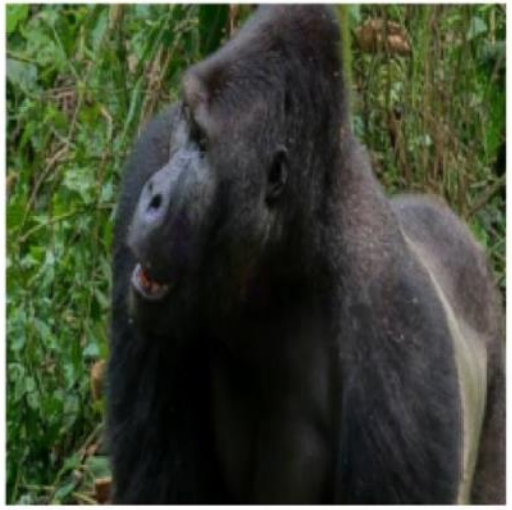
- The simplest form of adversarial attack
- Noise is a random arrangement of pixels containing no information
  - E.g., random numbers from a normal distribution (with 0 mean and 1 st.dev.)
- This attack represents a non-targeted black-box evasion attack
  - It is not efficient, since the noisy images are easily distinguishable from the original images



Evasion Attack: Semantic Attack

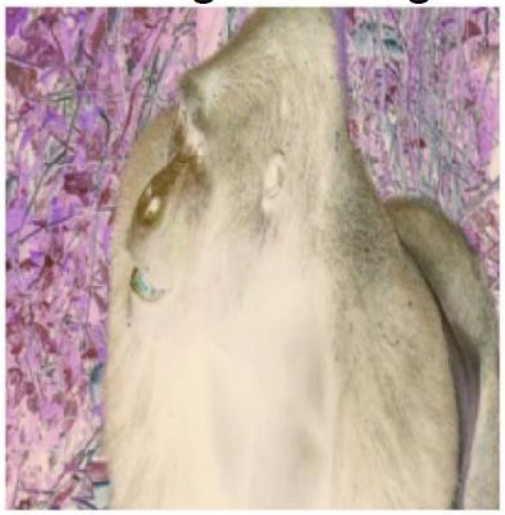
- Use negative images as adversarial inputs
  - Reverse all pixels intensities
  - E.g., change the sign of all pixels, if the pixels values are in range [-1,1]
- Not an efficient attack, since the adversarial images can easily be detected

Original image



Prediction: gorilla

Negative image

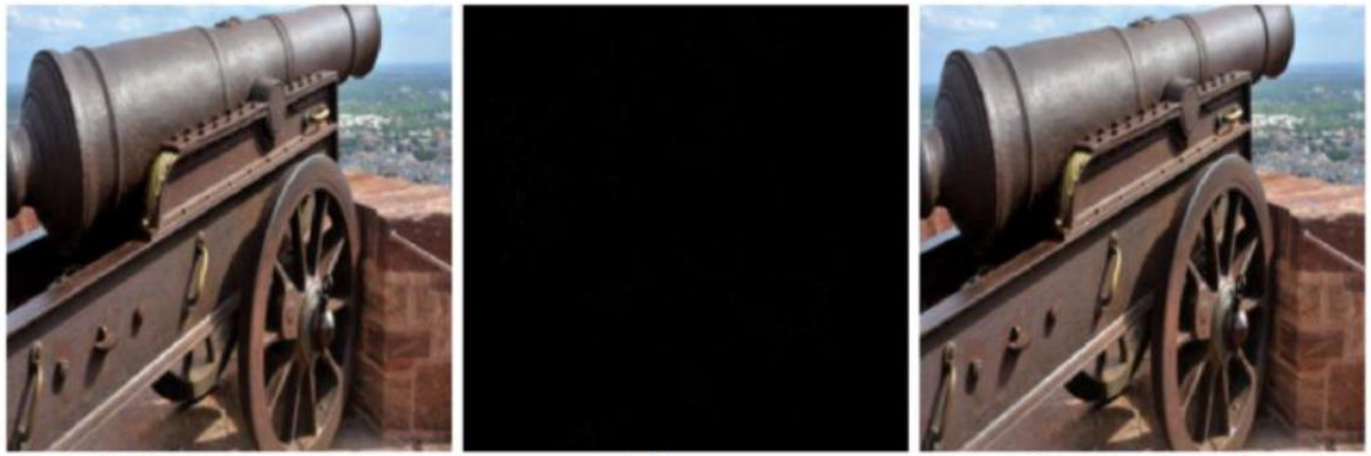


Prediction: weimaraner

Weimaraner (a dog breed)

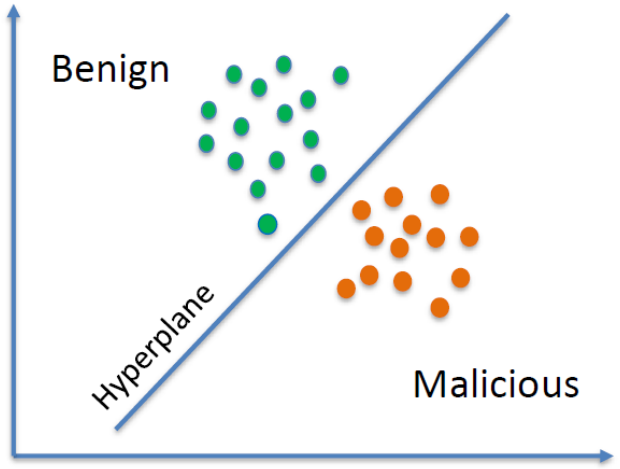
DeepFool Attack

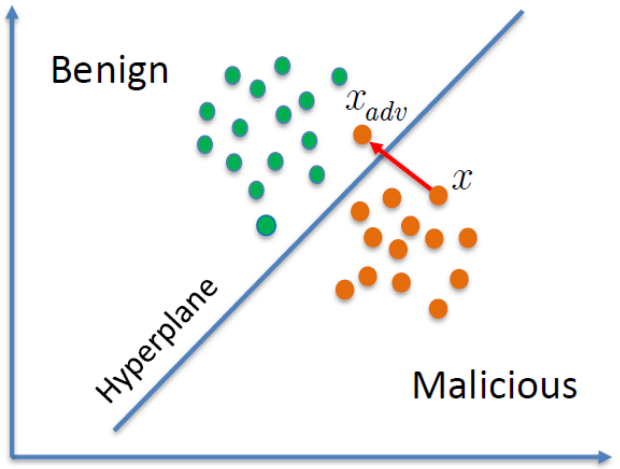
- DeepFool is a white-box attack
- It generates adversarial examples with the minimal amount of perturbation possible
- There is no visible change to the human eye between the left image (original sample) and the right image (adversarial sample)



DeepFool Attack

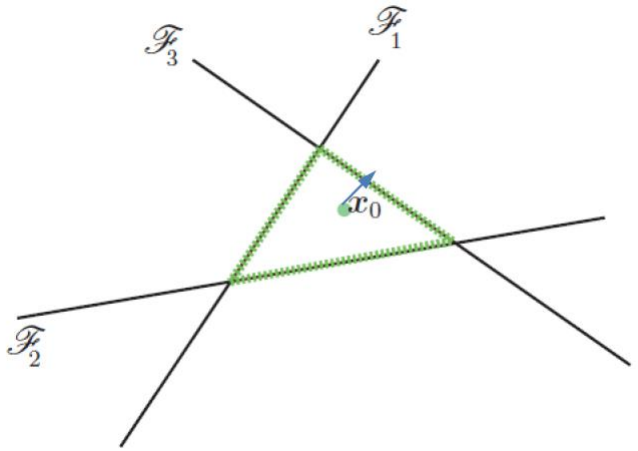
- For example, consider a linear classifier algorithm applied to objects from 2 classes: green and orange circles
- The line that separates the 2 classes is the hyperplane
  - Data points falling on either sides of the hyperplane are attributed to different classes (such as benign vs. malicious class)
- Given an input *x*, DeepFool projects *x* onto the hyperplane and pushes it just a bit beyond the hyperplane, thus misclassifying it





DeepFool Attack

- For a multiclass problem with linear classifiers, there are multiple hyperplanes that separate an input *x* from other classes
  - E.g., an example with 4 classes (0, 1, 2, and 3) is shown in the image below
- •DeepFool finds that closest hyperplane to the input *x*0: in this case, this is the hyperplane $\mathcal{F}3$ (it represents the most similar class to *x*0 of the other 3 classes)
- Then, it projects the input *x*0 onto the hyperplane $\mathcal{F}3$ and pushes it a little beyond it



Fast gradient sign method (FGSM) attack

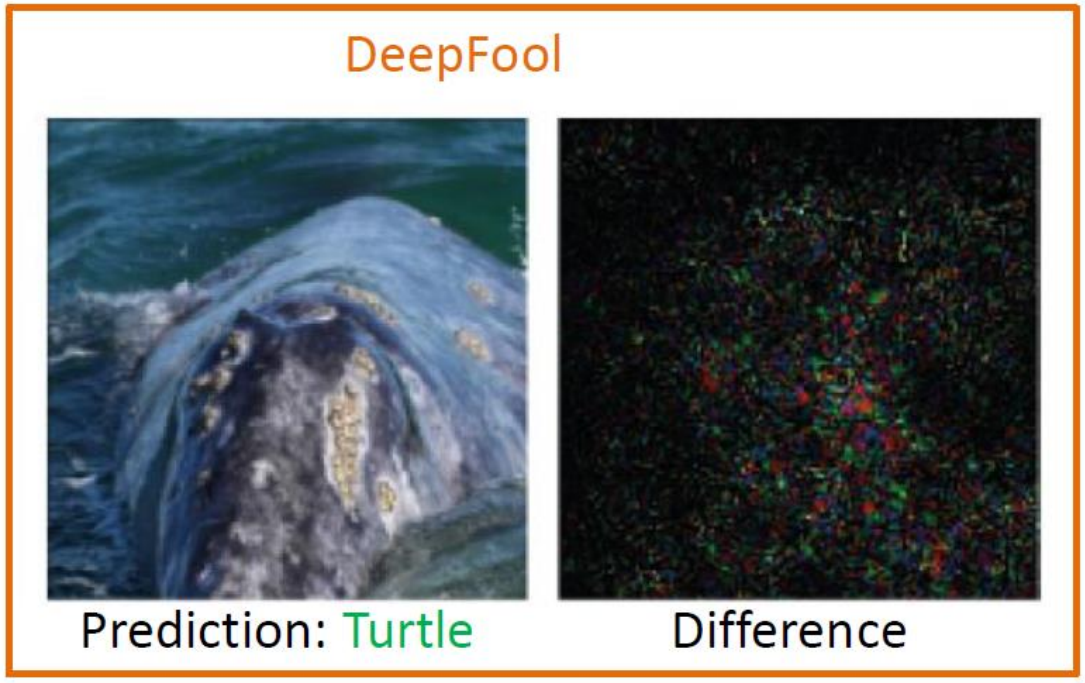
- An adversarial image x_{adv} is created by adding perturbation noise to an image x
- $x_{adv} = x + \epsilon \cdot \text{sign}(\nabla x \mathcal{L}(C(x,w),y))$
  - Notation: input image x, label y, NN model C, NN weights (parameters) w, loss function $\mathcal{L}$, gradient ∇(Greek letter "nabla"), perturbation magnitude ϵ
- The amount of perturbation is calculated based on the gradient of the loss function $\mathcal{L}$ with respect to the input image x for the true class label y.
- The perturbation increases the loss $\mathcal{L}$ for the true class y, and the model C misclassifies the image x_{adv}

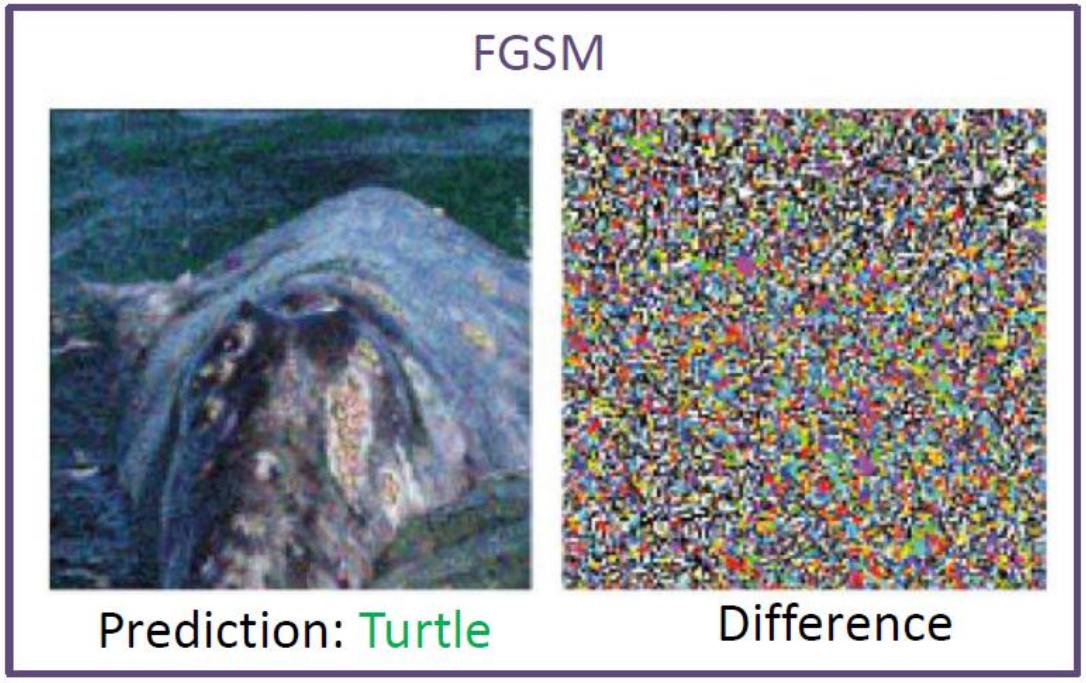
Fast gradient sign method (FGSM) attack

- FGSM is a white-box non-targeted evasion attack
- White-box, since we need to know the gradients $\nabla x\mathcal{L}(C(x,w),y)$ of the model to create the adversarial image
- In the shown example, the noise magnitude is ϵ= 0.007
  - Note: nematode is an insect referred to as roundworm
- FGSM calculates how much perturbation to add to each pixel in x, so that the loss $\mathcal{L}$ is maximized, leading to incorrect prediction by the model (i.e., $C(x,w) \neq y$)

DeepFool Vs FGSM

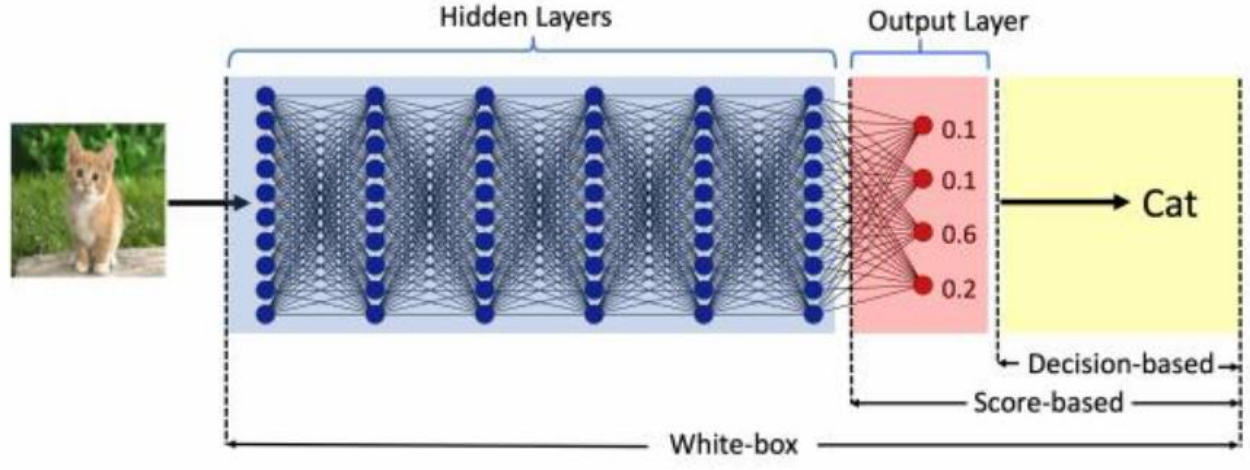
- Comparison of added adversarial perturbation for DeepFool and FGSM
  - Original image: whale
- Both DeepFool and FGSM perturb the image to be classified as turtle (targeted attack)
- DeepFool finds smaller perturbation





Black-box Evasion Attacks

- ***Query-based attacks***
  - The adversary queries the model and creates adversarial examples by using the provided information to queries
- The queried model can provide:
  - Output class probabilities (i.e., confidence scores per class) used with score-based attacks
  - Output class, used with decision-based attacks



Defense Against Evasion Attacks

- Adversarial samples can cause any ML algorithm to fail
- However, they can also be used to build more accurate and robust models
- The goal of ***adversarial defense*** is to build ML models with high accuracy on both clean (regular, natural, standard) examples and adversarial examples
- AML is a two-player game:
  - Attackers aim to produce strong adversarial examples that evade a model with high confidence while requiring only a small perturbation
  - Defenders aim to produce models that are robust to adversarial examples (the models either don't have adversarial examples, or the adversaries cannot find adversarial examples easily)

Defense Against Evasion Attacks

- Accordingly, the defenses can be categorized into:
  - Adversarial example detection
  - Gradient masking/obfuscation
  - Robust optimization

- ***Adversarial examples detection*** methods are designed to distinguish adversarial examples from regular clean examples
  - If the defense method detects that an input example is adversarial, the classifier will refuse to predict its class label
  - If an input example is classified as benign, then it is safe to be fed to the main classifier to predict its class

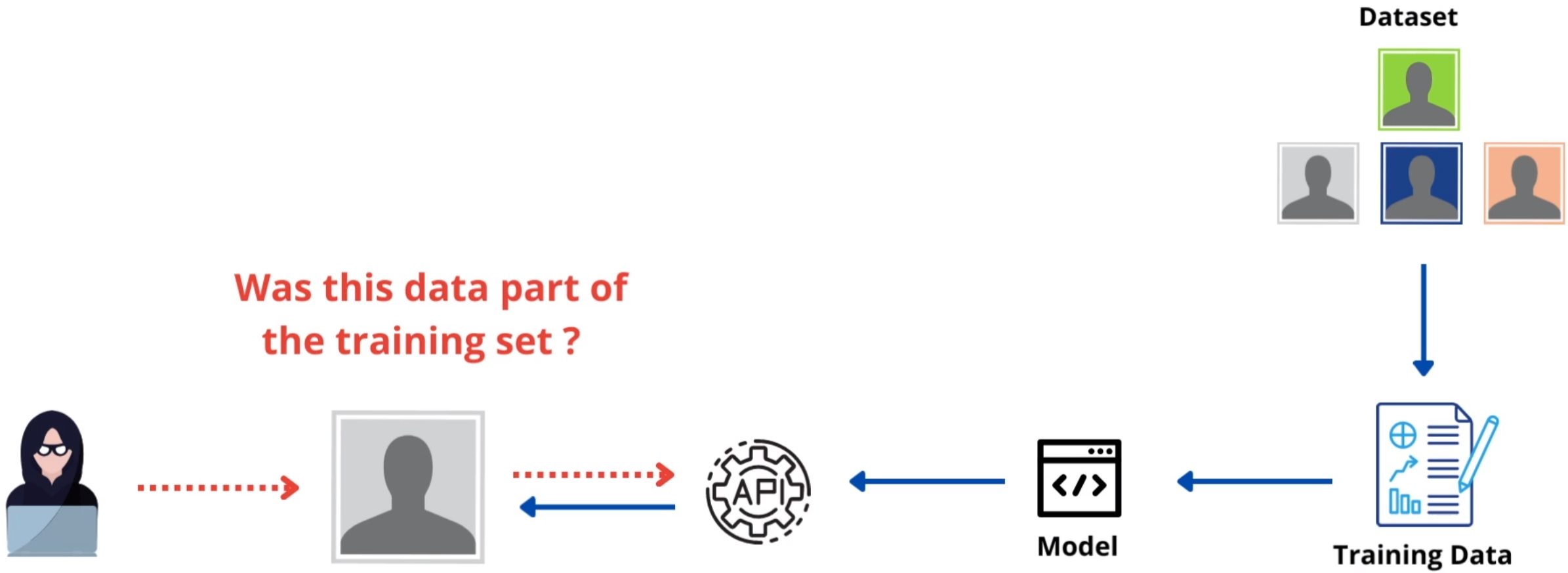
Adversarial Example Detection

- An auxiliary detection model is trained on regular and adversarial examples to perform a binary classification
  - An auxiliary NN uses input images for binary classification
- Statistical methods employs statistical tests for detecting adversarial samples
  - Detection based on the maximum mean discrepancy between the distributions of clean and adversarial samples
  - The distribution of principal components of inputs is used for detection
- Prediction consistency methods extract features that vary between clean and adversarial samples

Defense Against Evasion Attacks

- ***Gradient masking/obfuscation*** defense methods deliberately hide the gradient information of the model (from being used by an adversary)
- ***Robust optimization defense*** aims to evaluate and improve the robustness of the target classifier to adversarial attacks
- ***Adversarial training*** is a method of Robust optimization defense for training or retraining the target classification model using adversarial examples
  - The training dataset is augmented with adversarial examples produced by known types of attacks
    - E.g., for each clean example add one adversarial example to the training set
  - Adversarial training is one of the most common adversarial defense methods currently used in practice
  - Limitation of adversarial training is reduced accuracy on clean samples, known as *accuracy versus robustness trade-off*

MEMBERSHIP INFERENCE
(Data Extraction)



Privacy Attacks in AML

- ***Privacy attacks*** are also referred to as inference attacks or confidentiality attacks
- They can broadly be developed against:
  - Training data
    - E.g., reveal the identity of patients whose data was used for training a model
  - ML model
    - E.g., reveal the architecture and parameters of a model that is used by an insurance company for predicting insurance rates
    - E.g., reveal the model used by a financial institution for credit card approval
- Privacy attacks are commonly divided into the following main categories
  - Membership inference attack
  - Feature inference attack
  - Model extraction attack

Privacy Attacks in AML

- ***Membership inference attack***
  - Adversarial goal: determine whether or not an individual data instance $x*$is part of the training dataset $\mathcal{D}$for a model
  - The attack typically assumes black-box query access to the model
- ***Feature inference attack***
  - Adversarial goal: recreate certain features of data instances $x*$or statistical properties (such as class average of $x*$) of the training dataset $\mathcal{D}$for the model
  - Various attacks have been developed to either recover partial information about the training data (such as sensitive features of the dataset, or typical representatives for specific classes in the dataset) or full data samples

Privacy Attacks in AML

- ***Model extraction attack***
  - Adversarial goal: reconstruct an approximated model $f'(x)$of the target model $f(x)$
  - A.k.a. model inference attack
  - The approximated function $f'x$ will act as a substitute model and produce similar predicted outputs as the target model
  - The adversary has black-box query access to the model
  - The goal is to “steal” the model and use the substitute model for lunching other attacks, such as synthesis of adversarial examples, or membership inference attacks
  - Besides creating a substitute model, several works focused on recovering the hyperparameters of the model, such as the number of layers, optimization algorithm, activation types used, etc.

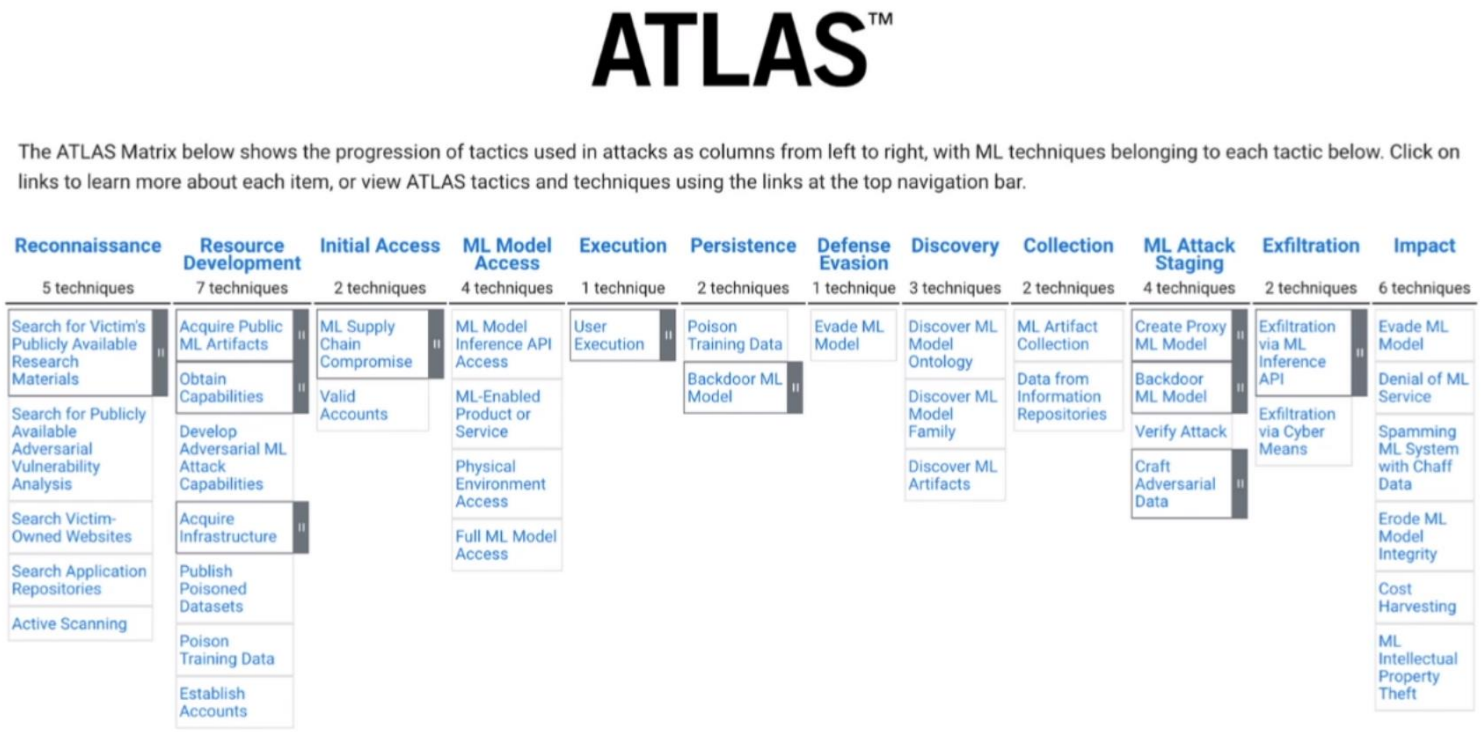
Risk Identified	Description	Security Control
Data Poisoning	Attacker alters data to modify the ML algorithm's behavior in a chosen direction e.g. poisoning the data fed to an algorithm to make it seem cats are dogs and vice versa or modifying facial recognition data	Data must be checked to ensure they will suit the model and limit the ingestion of malicious data. Check the trust level of the data source and protect the integrity of the data pipeline.
Model Poisoning	A type of attack in which the attacker has injected a rogue model in the AI lifecycle as a backdoor. This is especially risky if the company is not creating its model from scratch and relying on publicly available ones (think supply chain attacks)	Do not reuse models taken directly from the internet without checking them. Use models for which the threats are clearly identified and for which security controls exist.
Data Leakage	The Attacker compromises and is able to access the live data that is fed to the model in the fine-tuning phase or production phase.	Ensure that data pipeline is secure from un-authorized access. If third party data sources are being used then ensure their integrity is checked (Supply Chain attacks)

Risk Identified	Description	Security Control
Model compromise	This threat refers to the compromise of a component or developing tool of the ML application. e.g.: compromise of one of the open-source libraries used by the developers to implement the ML algorithm.	Verify that any software libraries used within your AI system do not have major vulnerabilities Define dashboards of key indicators integrating security indicators (peaks of change in model behavior etc.) in particular to allow rapid identification of anomalies.
Model Evasion	Attacker works on the ML algorithm's inputs to find small inputs leading to large modification of its outputs (e.g. decision errors). Basically trying to trick the ML algorithm to make wrong decisions	Generate adversarial examples for testing the AI model and attempt to perform an evasion attack. Make sure it is part of your testing suite

Risk Identified	Description	Security Control
Model compromise	This threat refers to the compromise of a component or developing tool of the ML application. e.g.: compromise of one of the open-source libraries used by the developers to implement the ML algorithm.	Verify that any software libraries used within your AI system do not have major vulnerabilities Define dashboards of key indicators integrating security indicators (peaks of change in model behavior etc.) in particular to allow rapid identification of anomalies.
Model Evasion	Attacker works on the ML algorithm's inputs to find small inputs leading to large modification of its outputs (e.g. decision errors). Basically trying to trick the ML algorithm to make wrong decisions	Generate adversarial examples for testing the AI model and attempt to perform an evasion attack. Make sure it is part of your testing suite

MITRE ATLAS (Adversarial Threat Landscape for Artificial-Intelligence Systems)

"a knowledge base of adversary tactics, techniques, and case studies for machine learning (ML) systems based on real-world observations, demonstrations from ML red teams and security groups, and the state of the possible from academic research"



- https://atlas.mitre.org/