Adversarial Machine Learning

CS463/ECE424

University of Illinois

Overview

- Attack Models and Taxonomy
- Case Studies
  - Spam Classification
  - Anomaly Detection
  - Face Authentication

Threat Models

- What are the capabilities of the attacker?
  - Can the attacker **influence the model training**?
  - Does the attacker **know the model parameters**?
  - Can the attacker **observe the output of the model on new instances**?
- What is the goal of the attacker?
  - To **avoid detection** of attacks?
  - To **cause benign input to be (mis)classified** as attack input?
  - To **launch targeted attacks or DoS**?

Taxonomy (1) Influence

Categories based on **influence**

- **Causative attacks** alter the training process through influence over the training data (poisoning, backdoor attacks)
  - Increase False Positives
  - Increase False Negatives

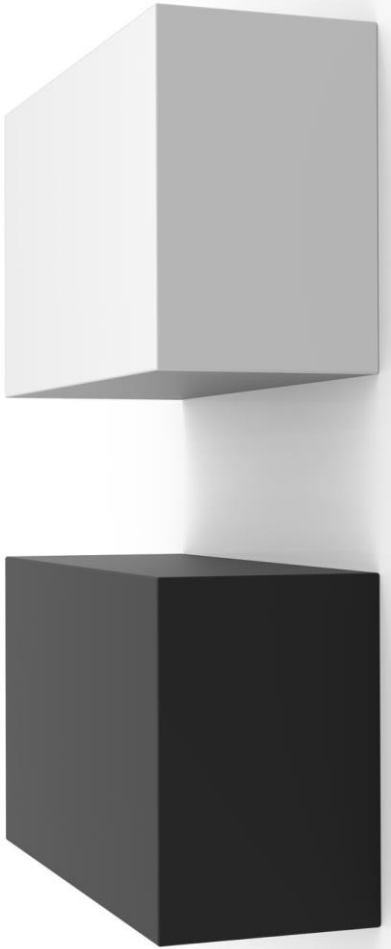
- **Exploratory attacks** do not alter the training process but use other techniques, such as probing the detector, to discover information about it or its training data (evasion, privacy attacks)

Taxonomy (2) Background Knowle

Categories based on **background knowledge**

- In **white-box attacks**, the adversary has access to the machine learning model (i.e., the model architecture and model parameters)
  - Useful for worst-case analysis

- In **black-box attacks**, the adversary only has access to the prediction APIs of the model
  - Understand the average-case



Taxonomy (3) – Security Violation

Categories based on **security violation**

- **Integrity** attacks result in intrusion points being classified as normal (i.e., cause false negatives)
- **Availability** attacks cause so many classification errors (e.g., false positives), that the system becomes effectively unusable
- **Privacy** violation: the adversary obtains information from the learner, compromising the secrecy or privacy of the system's users

Taxonomy (4) – Specificity

Categories based on **specificity**

- In a **targeted** attack, the focus is on a single or small set of target samples
- An **indiscriminate** adversary (non-targeted) has a more flexible goal that involves a very general class of points, such as “any false negative”

Case Studies

Case Study: Spam Filtering

- Exploratory Integrity Attacks
  - Goal: get modified spam messages into the user's inbox
  - Approach:
    - Include contents not indicative of spam
    - Modify spam like contents
- Causative Availability Attacks
  - Goal: denial of service: blocking benign emails
  - Indiscriminative dictionary attack
  - Targeted attack

Attack Model: Causative Availability Attacks

- Contamination Assumption (Poisoning):
  - The attacker can send emails that the victim will use for training
  - Attack emails are always trained as “spam” (i.e., by using bad IPs)
- Attack Cost:
  - The number of attack emails
- What should be included in the attack email?
  - More words or fewer?

Nelson, Blaine, et al. "Exploiting Machine Learning to Subvert Your Spam Filter." *LEET* 8 (2008): 1-9.

Dictionary Attack: Untargeted

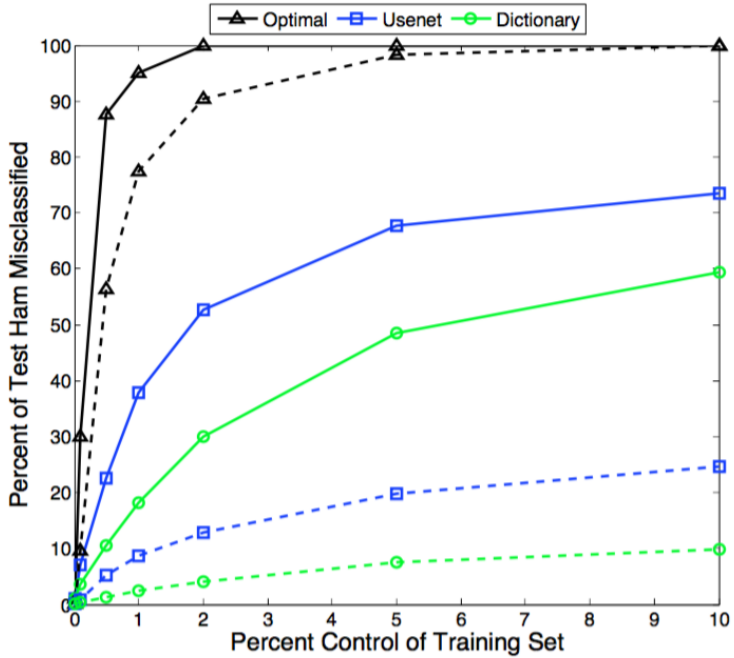
- Goal: make the spam filter unusable

Idea: adding "benign" words to spam emails, poisoning the training process

- **Optimal:** use all of the words
- **Usenet:** top ranked words from the Usenet corpus (90,000 common words)
- **Dictionary:** Aspell dictionary (98,568 misspellings and slang terms)

Dashed Line: ham classified as spam

Solid Line: ham classified as spam or unsure

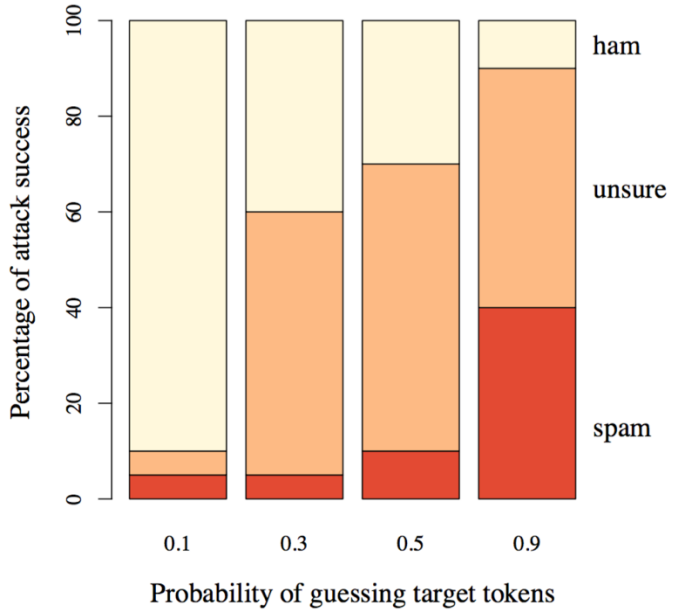


Targeted Attack

- Goal: To block target emails
  - E.g., only block emails with keyword “meeting”

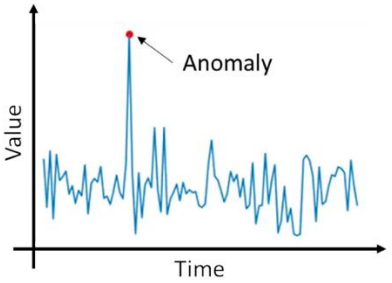
Assumption: Attacker has prior knowledge of the target emails, i.e., the keywords/tokens in the target emails

Idea: Inject “spam” emails with the target tokens



Poisoning Attack for Traffic Anomalies Detection

- Adversary's goals
  - Launch a DoS on some victim
  - Attack traffic must cross the network without detection



- Overview of the strategy: **poisoning (i.e., causative attack)!**
  - Add additional traffic called *chaff* over the targeted flow
  - Anomaly detector retrained periodically with recent (poisoned) traffic
  - The amount and the time period for which chaff is introduced depends on the adversary's knowledge of the network

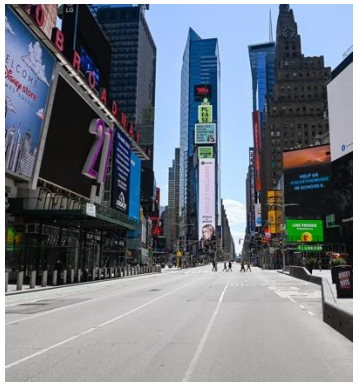
Attack Strategies

- t : time
- c_t: quantity of chaff to add to the target flow
- θ : attack parameter, controls the intensity of the attack

- **Un-informed Chaff Selection (Naïve baseline)**
  - Use Bernoulli Random variable to decide whether to add traffic
  - $c_t = \theta$ (i.e., constant volume)

Strategy 1: Locally Informed Chaff Selection

- **Locally Informed Chaff Selection**
  - Add-More-If-Bigger
  - Add chaff when the traffic volume on the link exceeds a parameter α
  - Define $y_s(t)$ as the volume of traffic in the ingress link the attacker controls
  - Add $c_t = (\max(0, y_s - \alpha))^{\theta}$



Strategy 2: Globally Informed Chaff Selection

- Assumptions:
  - The adversary has **complete knowledge** of all the traffic in the network, the detection algorithm A, and future measurements y_t
  - The adversary can introduce the chaff along any flow
    - This becomes a ”white-box” attack

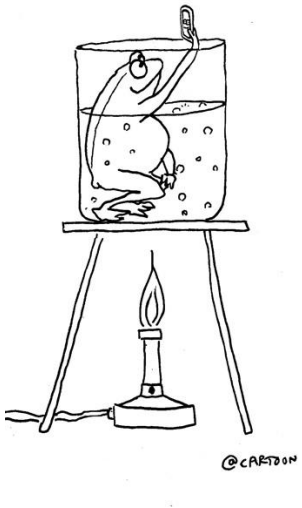
- Can be formalized as an optimization problem:
  - To maximize injected volume c_t, with the constraint that $A(y_t + c_t)$ cannot be detected

Attack Strategies under Different Assumptions

- Uniformed Chaff Selection
  - No knowledge on volume of traffic
  - $c_t = \theta$
- Locally Informed Chaff Selection
  - Knowledge of the traffic volume **on a single link**
  - $c_t = (\max(0, y_s - \alpha))^{\theta}$
- Globally Informed Chaff Selection
  - Complete knowledge of traffic volume
  - Optimization problem
- **How to choose θ ?**

Boiling Frog Poisoning

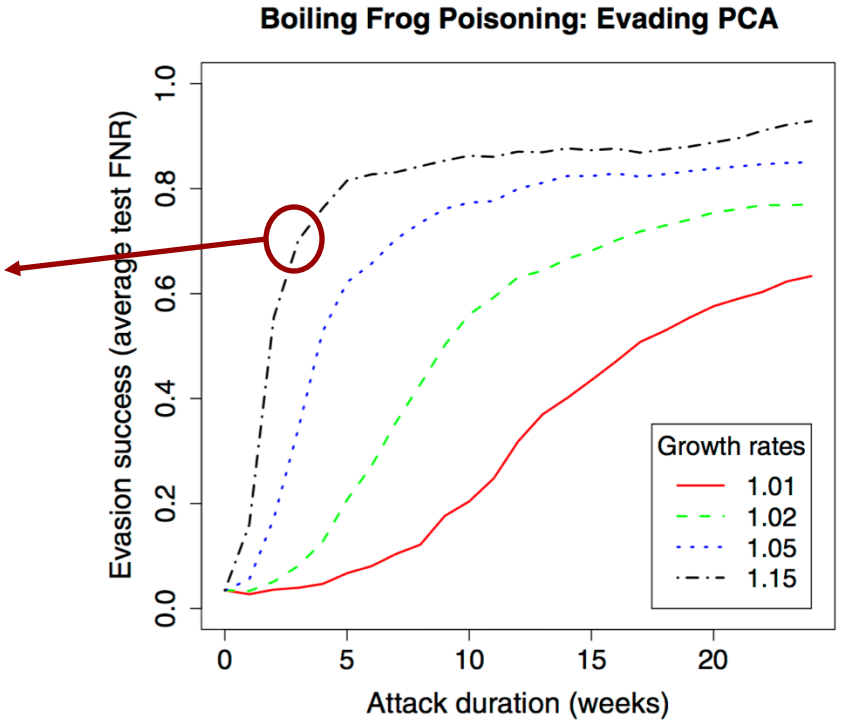
- Initially set θ to a **small value,** increase it **slowly** over time
  - Add additional malicious training data each successive week
  - Detection model is retrained every week
- The **training data contains malicious events** that were not detected by the previous detector
- Repeat this until the week of attack



Key Idea: increasing the threshold of the target detector gradually

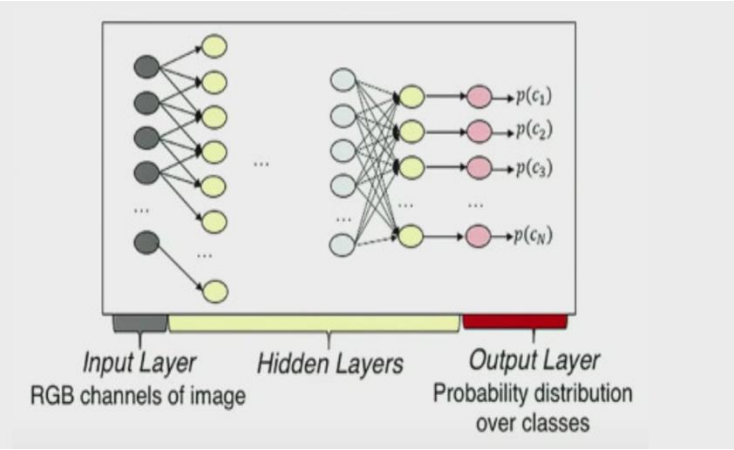
Boiling Frog Poisoning

Blackline: with a 15% growth rate, the FNR is increased to 70% over 3 weeks of poisoning



Deep Neural Network (DNN)

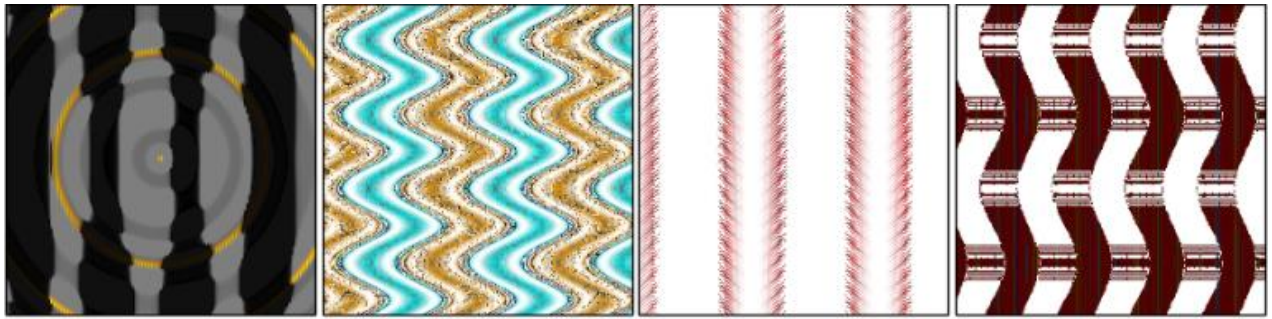
- Convolutional neural network (CNN)
- Layers of neurons (basic computation unit)
- Good at image recognition



Adversarial Examples

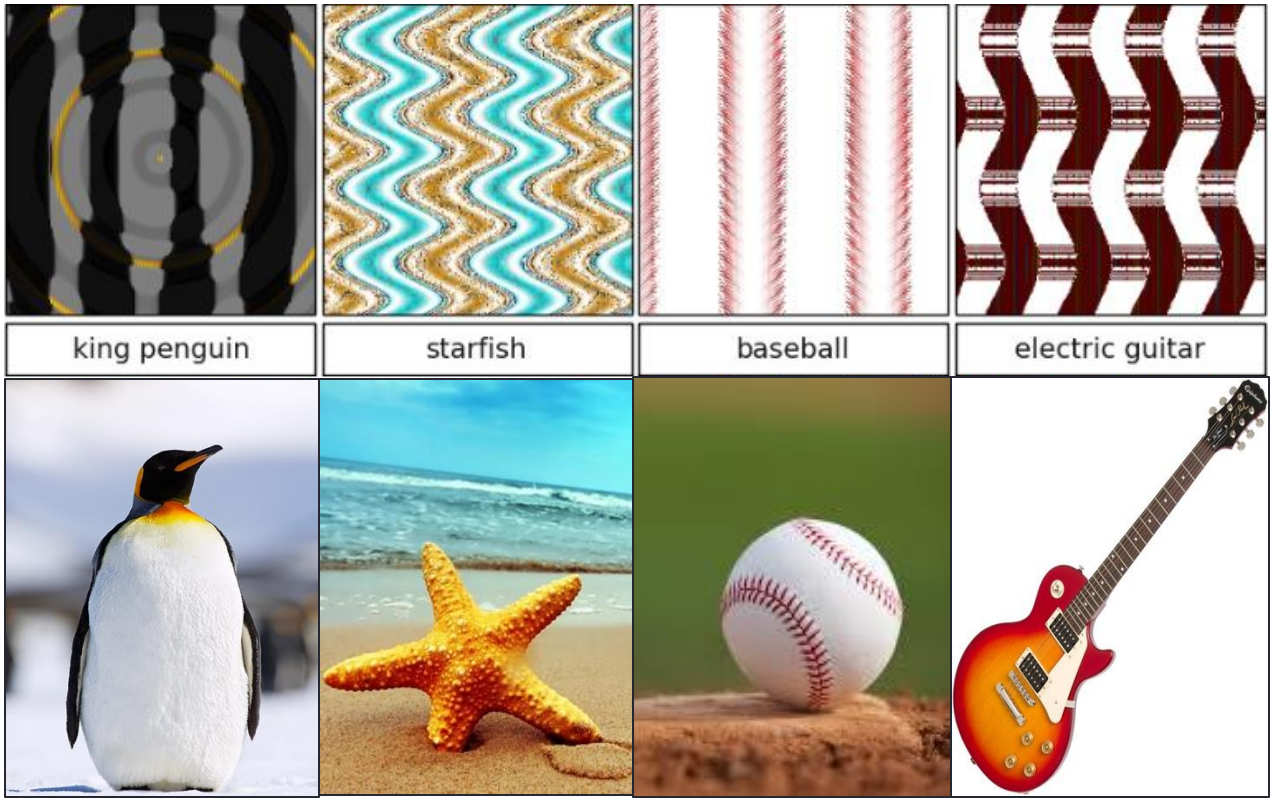
Adversarial Examples are synthetic examples constructed by modifying real examples slightly in order to make a classifier believe they belong to the wrong class with high confidence.

Adversarial Examples



DNN “sees” pictures differently from humans.

Adversarial Examples

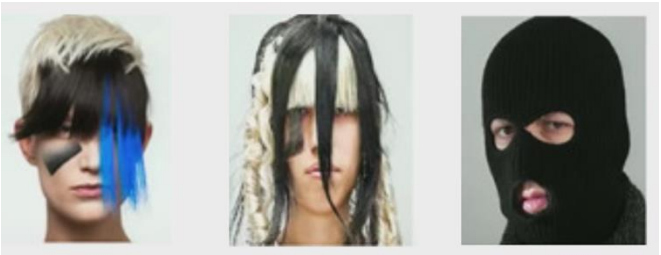


Adversarial Examples

Ostrich!

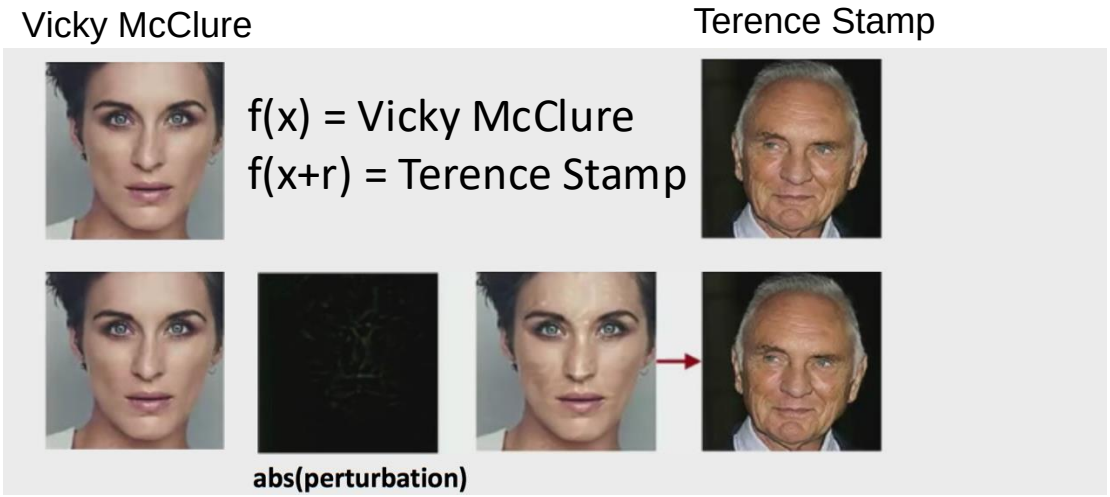
Attack on DNN Face Recognition

- **Goal:** To impersonate a victim in the real world
- Challenges:
  - **Physical Realizability**
    - The attacker can only change his own appearance
    - Robust to changes in different imaging conditions
  - **Inconspicuousness**
    - Do not raise too much of suspicion
    - Avoid physical appearances like



Sharif, Mahmood, et al. "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition." *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016.

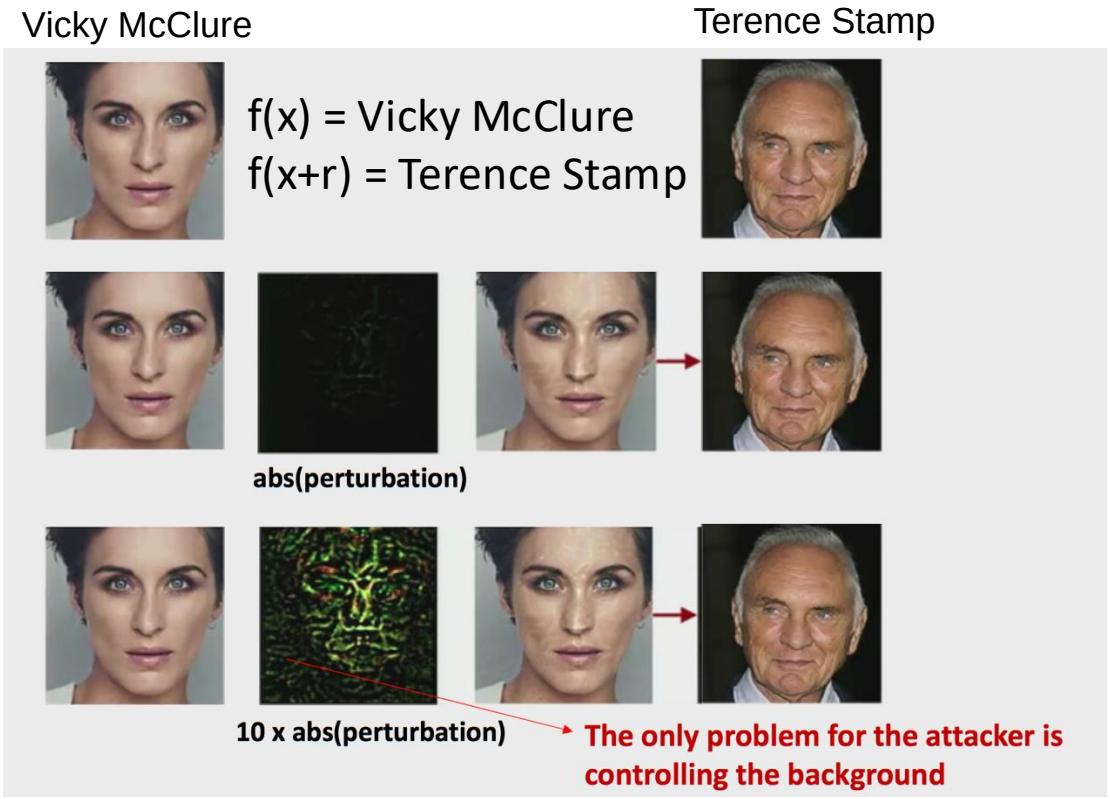
Impersonation Attack



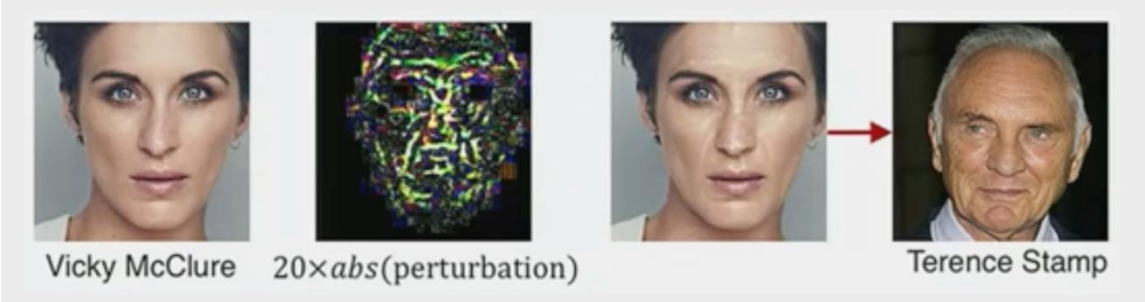
$$\underset{r}{\operatorname{argmin}}\ softmaxloss(f(x+r), c_t)$$

Perturbation Target Class

Impersonation Attack



Apply Changes to the Face Only



- Step 1: Face detection
- Step 2: Only change pixels that overlay the face

$$\underset{r}{\operatorname{argmin}}\ softmaxloss(f(x+r), c_t)$$

Perturbation

Target Class

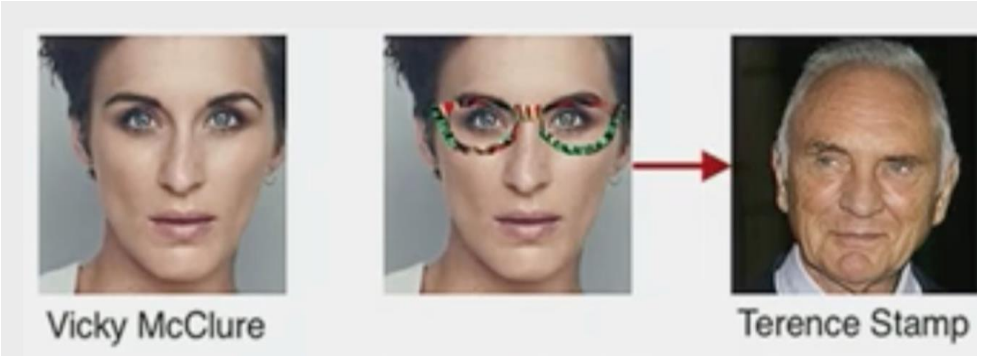
How to realize the attack: we need something…

- Easy to produce
- Overlaying the face
- Not associated with adversarial intent

Apply Changes to the Eyeglasses

How to realize the attack: we need something…

- Easy to produce
- Overlaying the face
- Not associated with adversarial intent



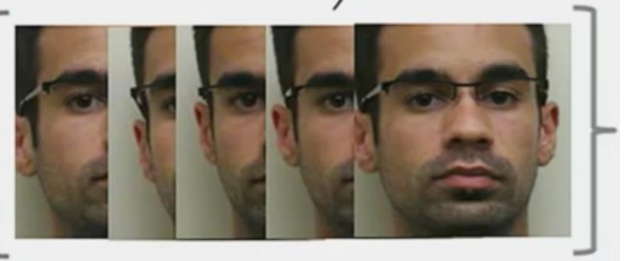
Success rate: 92%

Robustness

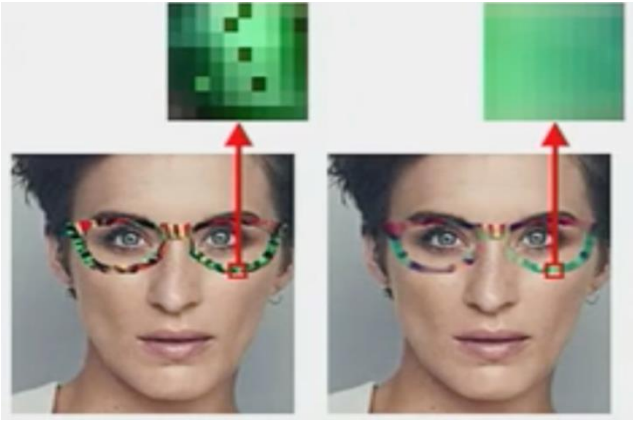
- Images of the same face are unlikely to be exactly the same
- The attack needs to work for **many** images of the adversary's face

$$\underset{r}{\operatorname{argmin}} \left(\sum_{x \in X} \text{distance}(f(x+r), c_t) \right)$$

X is a set of images, e.g., $X =$



Smooth Transitions



Can cameras correctly capture the perturbation?

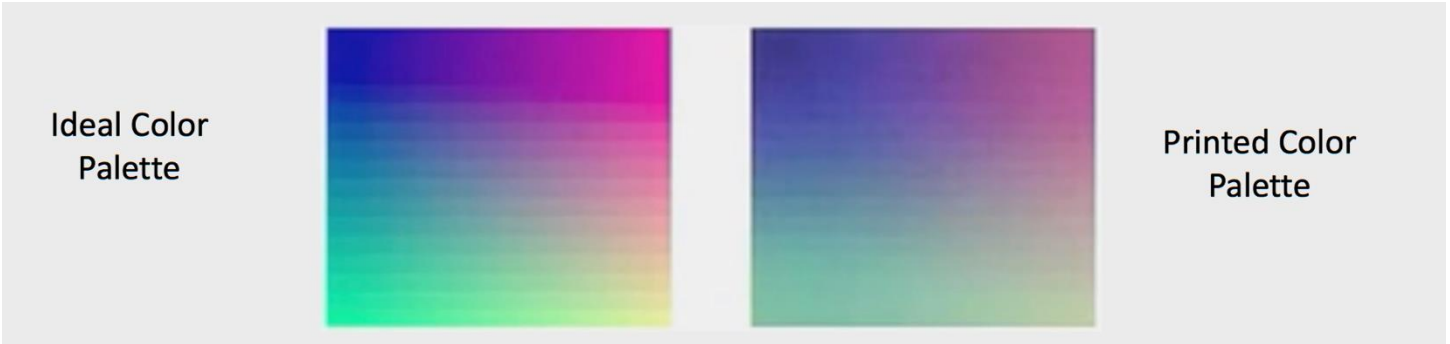
- Natural images are **smooth**
- Distances between neighboring pixels are small
- Measured by **total variation** (TV)

$$TV(r) = \sum_{i,j} \left((r_{i,j} - r_{i+1,j})^2 + (r_{i,j} - r_{i,j+1})^2 \right)^{\frac{1}{2}}$$

Printability

- The range of colors that a printer can reproduce is a **subset** of RGB values
- **Non-printability score (NPS)**

$$NPS(\hat{p}) = \prod_{p \in P} |\hat{p} - p|$$



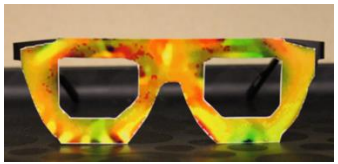
Physical Realizable Impersonation

- An optimization problem

$$\underset{r}{\operatorname{argmin}} \left(\sum_{x \in X} \text{distance}(f(x+r), c_t) \right) + \kappa_1 \cdot \text{TV}(r) + \kappa_2 \cdot \text{NPS}(r)$$

misclassify as c_t (set of images) — smoothness — printability

- Attack steps:
  - Choose a DNN model and a target to impersonate
  - Get a set of photos of the attacker (X)
  - Calculate the perturbation
  - Print the glasses



Experiment

- **Procedure:**
  - Collect images of attacker
  - Choose random target
  - Generate and print eyeglasses
  - Collect 30 to 50 images of attacker wearing the eyeglasses
  - Classify the collected images
- **Success metric:** the fraction of collected images that are misclassified as target

Experiment

S_A

Success Rate: 87.87%
Mean Probability: 0.78

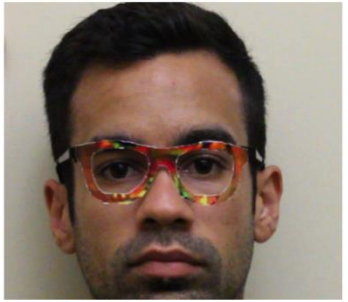
Milla Jovovich

S_B

Success Rate: 88%
Mean Probability: 0.75

S_C

S_C

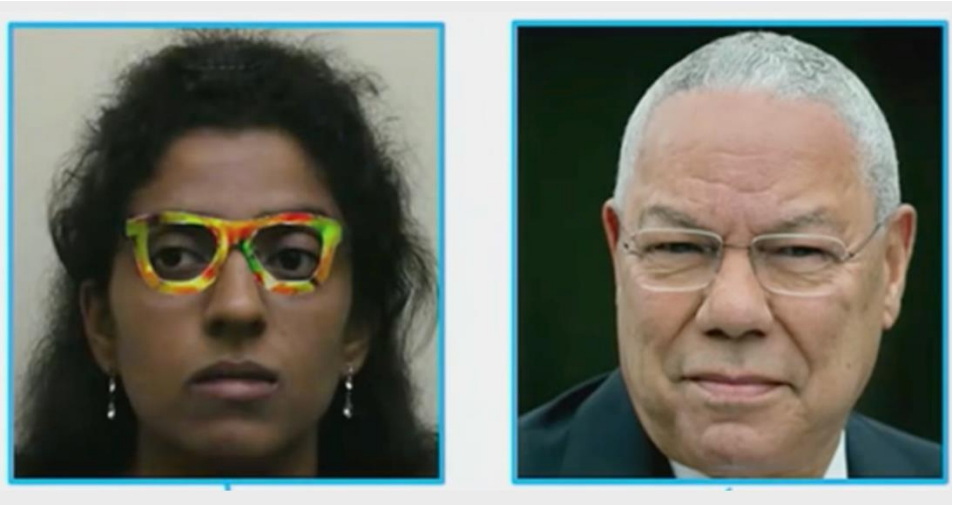


Success Rate: 100%
Mean Probability: 0.99

Carson Daly

Limitations

- Low success rate for some targets



Colin Powell, **16% success rate**

- Small variations in lighting

Challenges in Defending Against Adversarial Learning

- A good defense requires machine learning models to produce good outputs **for every possible input**.

- Lack of (reliable) theoretical model
  - Lack of theoretical understanding of **the machine learning models** (especially neural networks)

Discussion Questions

- How can we defend against the adversarial machine learning attacks mentioned in this lecture? (Pick one attack to discuss)
  - Dictionary attacks on spam filtering
  - Poisoning attacks on anomaly detection
  - Impersonation attacks on DNN Face Recognition

- The CIA security principles help us design secure systems. Do you think we can apply these principles in designing secure machine learning models? Can you think of other principles for designing secure machine learning models?

Reading

- [1] Huang, Ling, et al. "Adversarial machine learning." *Proceedings of the 4th ACM workshop on Security and artificial intelligence*. ACM, 2011.
- [2] Rubinstein, Benjamin IP, et al. "Antidote: understanding and defending against poisoning of anomaly detectors." *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*. ACM, 2009.
- [3] Sharif, Mahmood, et al. "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition." *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016.