Notes

- Call for actions
  - [Important!] Switch to online from the next class (11/07)
  - [Important!] Checkpoint Presentation II (online, on 11/07)
    - 12-min presentation + 3 min Q&A
    - Presentation MUST cover:
      - 1 slide on your research topic
      - 1-2 slides on your goals and ideas (how do you plan to achieve your goals)
      - 1-2 slides on your *experimental design*
      - 1-2 slides on your *preliminary results* [*very important*]
      - 1 slide on your *next steps* until the final presentation

CS 499/579: Trustworthy ML
Targeted poisoning attacks

Tu/Th 4:00 – 5:50 PM

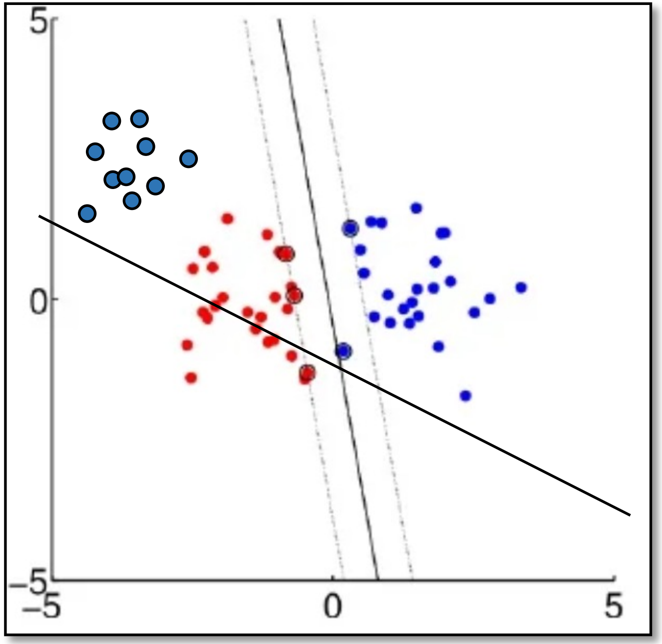
Sanghyun Hong

sanghyun.hong@oregonstate.edu

Oregon State University

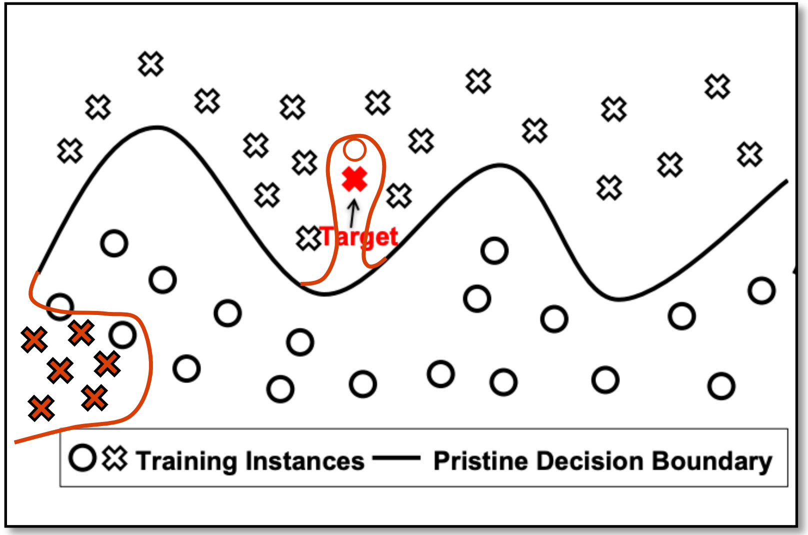
SAIL
Secure AI Systems Lab

Recap: conceptual illustration of the vulnerability to poisoning



← Linear model (SVM)

Neural Network →



Targeted poisoning threat model

- Goal
  - **Targeted** attack
  - Model causes a misclassification of (x_t, y_t), while preserving acc. on D_{val}
- Capability
  - Know a target (x_t, y_t)
  - Pick p candidates from test data $(x_{c1}, y_{c1}), (x_{c2}$… and craft poisons $(x_{p1}, y_{p1}), (x_{p2}$…
  - Inject them into the training data
- Knowledge
  - D_{tr} : training data
  - D_{test}: test-set data (validation data)
  - f: a model and its parameters θ
  - A: training algorithm (*e.g.*, mini-batch SGD)

Targeted poisoning threat model

- Goal
  - Targeted **clean-label** ($y_{c1} = y_{p1}$) attack
  - Model causes a misclassification of (x_t, y_t), while preserving acc. on D_{val}
- Capability
  - Know a target (x_t, y_t)
  - Pick p candidates from test data (x_{c1}, y_{c1}), $(x_{c2}$… and craft poisons (x_{p1}, y_{p1}), $(x_{p2}$…
  - Inject them into the training data
- Knowledge
  - D_{tr} : training data
  - D_{test}: test-set data (validation data)
  - f: a model and its parameters θ
  - A: training algorithm (*e.g.*, mini-batch SGD)

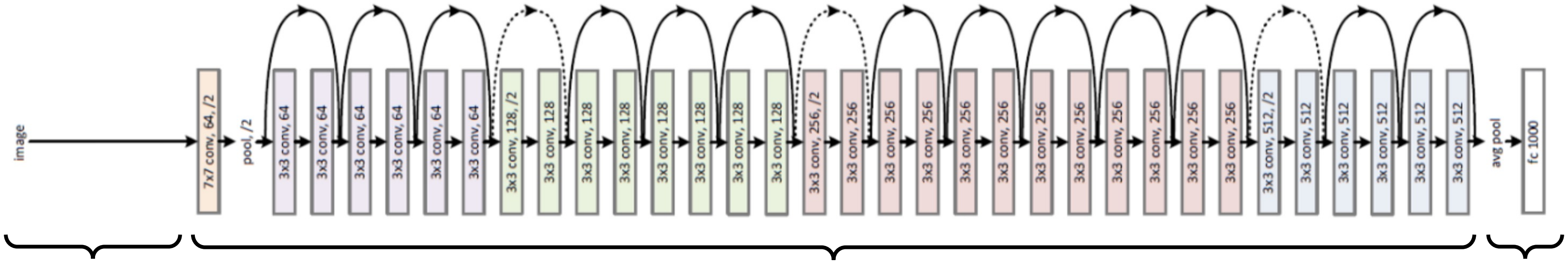
Topics for part II – data poisoning

- Research questions
  - What are some examples of poisoning attacks?
  - How can we generate *indiscriminate* poisoning examples?
  - How can we synthesize poisoning samples for *targeted* attacks?
  - How can we mitigate data poisoning attacks?

How can we perform clean-label targeted attacks?

Poison frogs! Targeted clean-label poisoning attacks on neural networks, Shafahi et al., NeuRIPS 2018

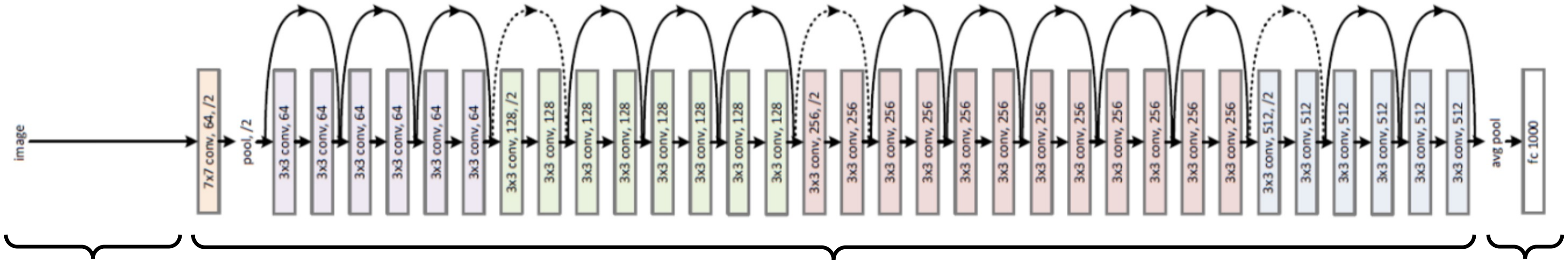
Background: convolutional neural networks



Input x Feature extractor: $f(\cdot)$ Classifier

- A conventional view:
  - Convolutions: extract features, embeddings, latent representations, …
  - Last layer: uses the output for a classification task

Background: convolutional neural networks

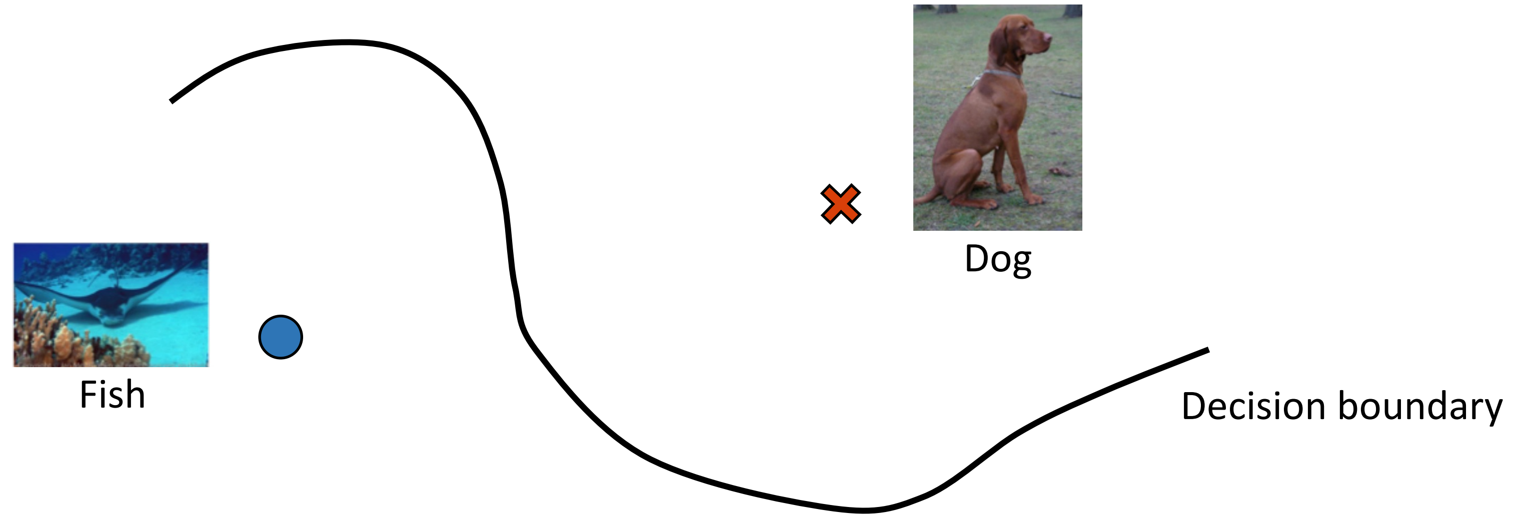


Input x | Feature extractor: $f(\cdot)$ | Classifier

- Input-space $\neq$ Feature-space:
  - Two samples similar in the input-space can be far from each other in the feature-space
  - Two samples very different in the input-space can be close to each other in f

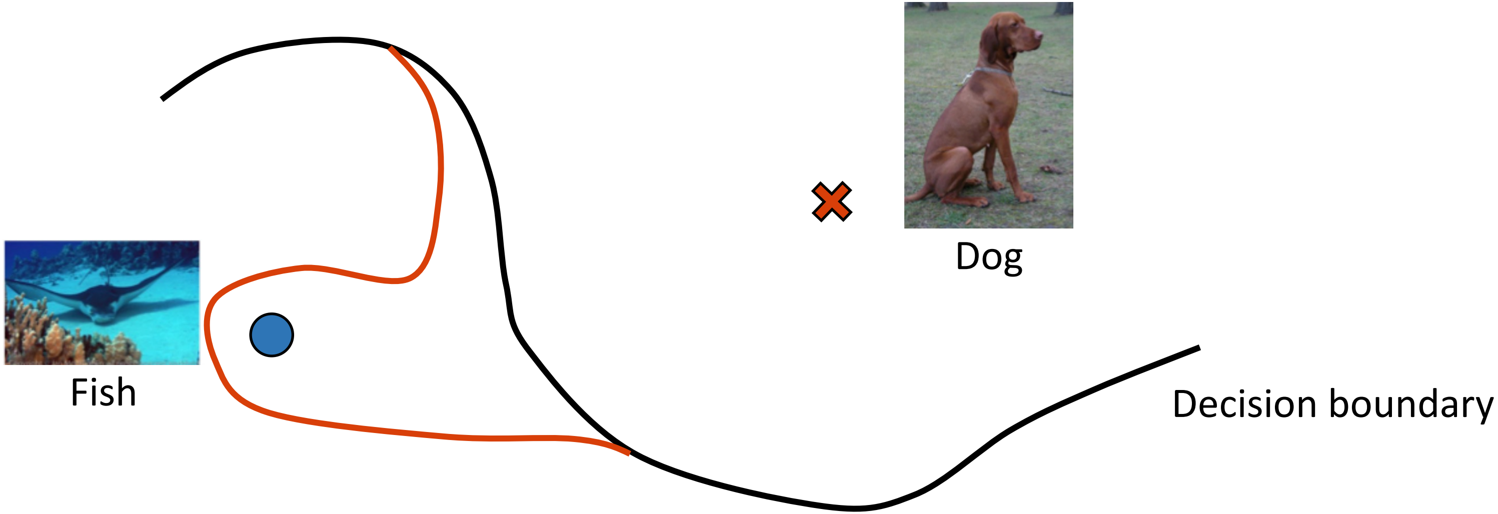
The key idea: feature collision

- Goal
  - You want your *any* poison to be closer to your target (x_t, y_t) in the *feature space*



The key idea: feature collision

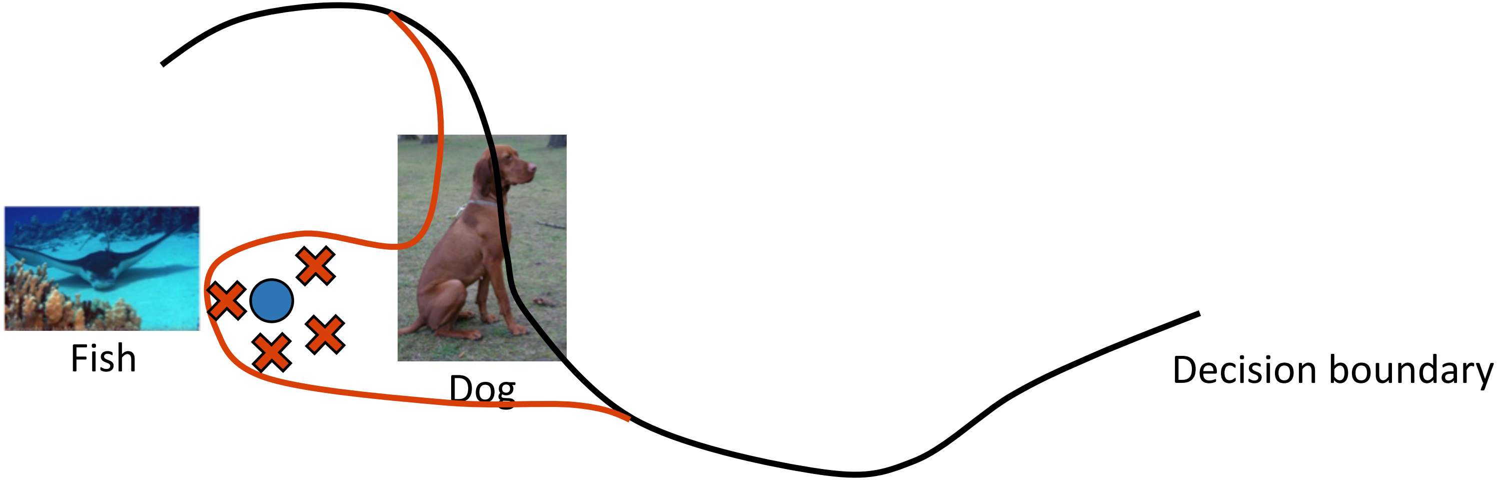
- Goal
  - You want your *any* poison to be closer to your target (x_t, y_t) in the *feature space*



The Fish Becomes DogFish!

The key idea: feature collision

- Goal
  - You want your *any* poison to be closer to your target (x_t, y_t) in the *feature space*



The key idea: feature collision

- Goal
  - You want your *any* poison to be closer to your target (x_t, y_t) in the *feature space*
  - Objective:

$$\mathbf{p} = \underset{\mathbf{x}}{\operatorname{argmin}} \; \|f(\mathbf{x}) - f(\mathbf{t})\|_2^2 + \beta \|\mathbf{x} - \mathbf{b}\|_2^2$$

  - Optimization:

Algorithm 1 Poisoning Example Generation

Input: target instance t, base instance b, learning rate λ
Initialize x: $x_0 \leftarrow b$
Define: $L_p(x) = \|f(\mathbf{x}) - f(\mathbf{t})\|^2$
for $i = 1$ **to** $maxIters$ **do**
  Forward step: $\widehat{x_i} = x_{i-1} - \lambda \nabla_x L_p(x_{i-1})$ // construct input perturbations
  Backward step: $x_i = (\widehat{x_i} + \lambda\beta b)/(1 + \beta\lambda)$ // decide how much we will perturb
end for

Evaluations

- Scenarios
  - Scenario 1: Transfer learning
  - Scenario 2: End-to-end learning

Evaluations: transfer learning

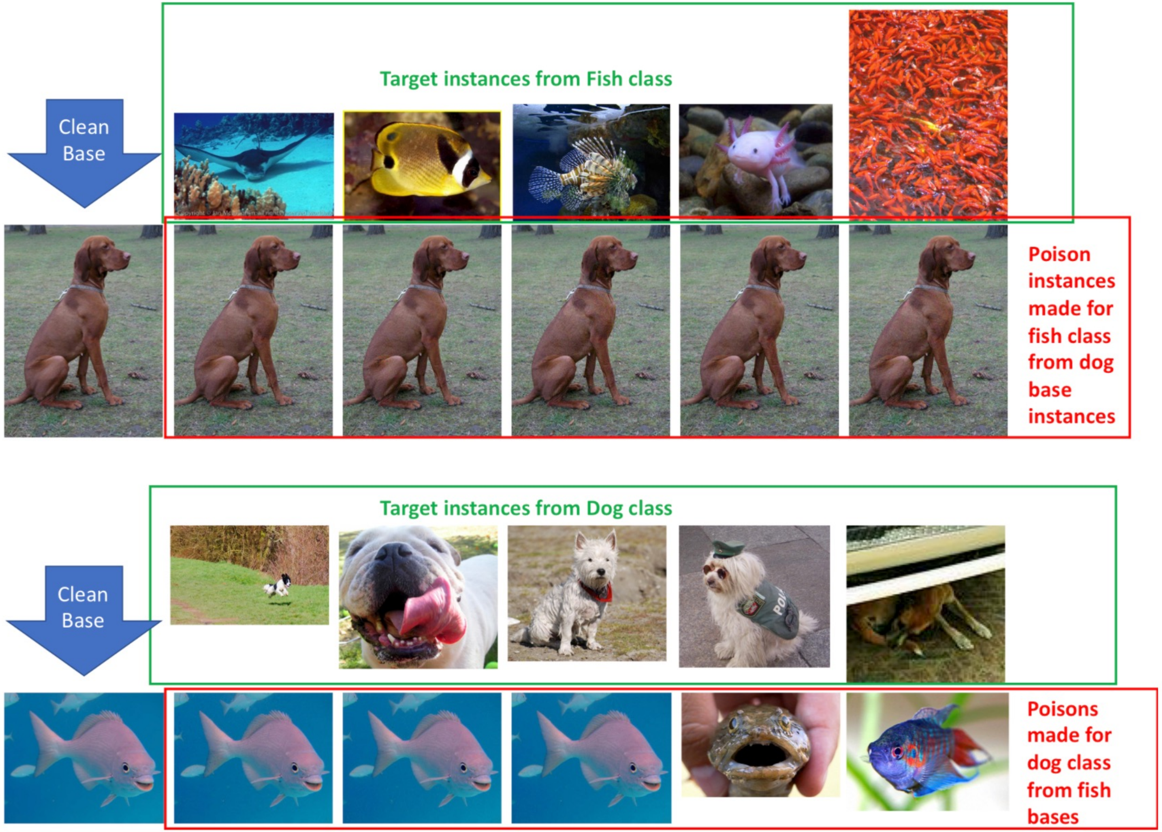
- Setup
  - Dataset: Dog vs. Fish (ImageNet)
  - Models: Inception-V3 (Pretrained on ImageNet)

- "one-shot kill" Attacks
  - Goal: Dog > Fish or Fish > Dog | All 1099 targets from the test-set
  - Craft a poison using a single image chosen from the other class
  - Train the last layer on $D_{tr} \cup (x_p, y_p)$ and check if the target's label is flipped

- Results
  - The attack succeeds with 100% accuracy
  - The accuracy drop caused by the attack is 0.2% on average

Evaluations: transfer learning

- Examples



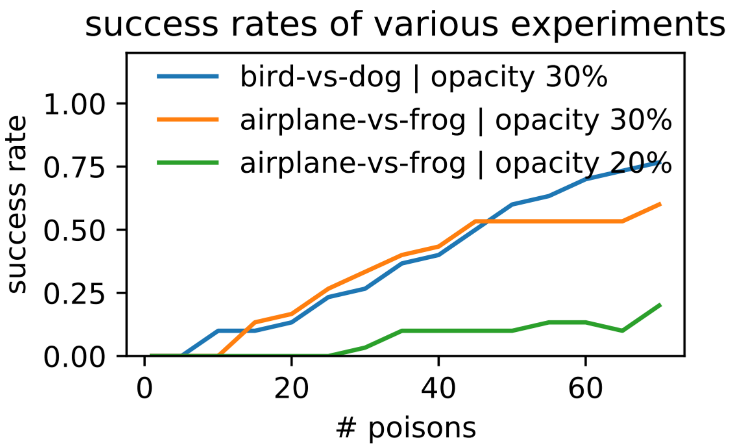
Evaluations: end-to-end learning

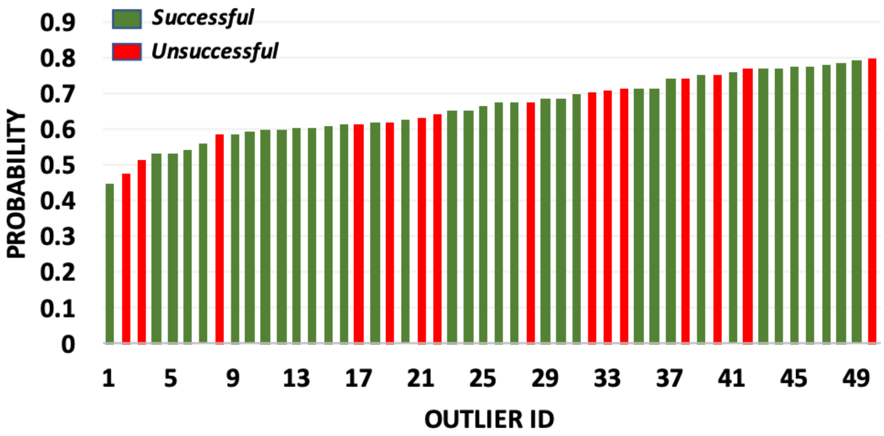
- Setup
  - Dataset: CIFAR-10
  - Models: AlexNet (Pretrained on CIFAR-10)

- "end-to-end" Attacks
  - Goal: Bird > Dog or Airplane > Frog
  - Craft 1-70 poisons using the images chosen from the (Dog or Frog) class
  - Trick: watermarking!
  - Train the entire model on $D_{tr} \cup (x_p, y_p)$ and check the misclassification rate

Evaluations: end-to-end learning

- Results





How can we improve the transferability of clean-label att.?

MetaPoison! Practical general-purpose clean-label data poisoning, Huang et al., NeuRIPS 2020

Revisit: Poisoning Threat Model

- Goal
  - Targeted **clean-label** ($y_{c1} = y_{p1}$) attack
  - Model causes a misclassification of (x_t, y_t), while preserving acc. on D_{val}

- Capability
  - Know a target (x_t, y_t)
  - Pick p candidates from test data (x_{c1}, y_{c1}), $(x_{c2}$… and craft poisons (x_{p1}, y_{p1}), $(x_{p2}$…
  - Inject them into the training data

- Knowledge
  - D_{tr} : training data
  - D_{test}: test-set data (validation data)
  - f: a model and its parameters θ
  - A: training algorithm (*e.g.*, mini-batch SGD)

Revisit: the key idea – feature collision

- Goal
  - Your poisons should work against any f and θ
  - Objective:

$$\mathbf{p} = \underset{\mathbf{x}}{\operatorname{argmin}} \; \underbrace{\|f(\mathbf{x}) - f(\mathbf{t})\|}_{\text{Now you don't know the } f\text{, how can you estimate this?}}{}_2^2 + \beta \|\mathbf{x} - \mathbf{b}\|_2^2$$

- Revisit the previous idea
  - Bi-level optimization

$$\begin{aligned} \arg\max_{\mathcal{D}_p} \quad & \mathcal{W}(\mathcal{D}', \boldsymbol{\theta}_p^\star), \\ \text{s.t.} \quad & \boldsymbol{\theta}_p^\star \in \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{D}_{tr} \cup \mathcal{D}_p, \boldsymbol{\theta}) \end{aligned}$$

$$\underbrace{\begin{aligned} X_p^* &= \underset{X_p}{\operatorname{argmin}} \; \mathcal{L}_{\text{adv}}(x_t, y_{\text{adv}}; \theta^*(X_p)) \\ \theta^*(X_p) &= \underset{\theta}{\operatorname{argmin}} \; \mathcal{L}_{\text{train}}(X_c \cup X_p, Y; \theta) \end{aligned}}_{\text{Problem: no control over } \theta}$$

The challenge: learning process

- Mode parameters are not fixed!
  - Initialization
  - Mini-batch-ed data
  - # of training epochs

Algorithm

Input: Examples $\{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C.

Initialize θ_0 randomly

for $t \in [T]$ **do**

  Compute gradient

  For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$

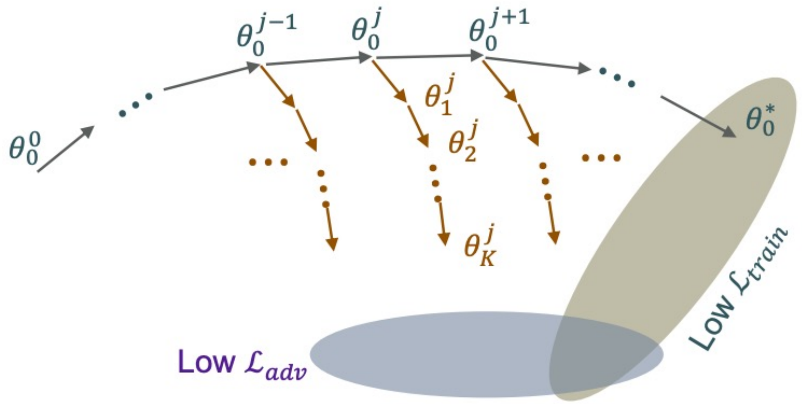
  Descent

  $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

Output θ_T and compute the overall privacy cost (ε, δ) using a privacy accounting method.

The key idea: unrolling

- Goal
  - You *simulate all* the training procedures with *possible* f, θs while crafting your poisons



Algorithm 1 Craft poison examples via MetaPoison

Input Training set of images and labels (X, Y) of size N, target image x_t, adversarial class y_{adv}, ϵ and ϵ_c thresholds, $n \ll N$ subset of images to be poisoned, T range of training epochs, M randomly initialized models.
Begin
Stagger the M models, training the mth model weights θ_m up to $\lfloor mT/M \rfloor$ epochs
Select n images from the training set to be poisoned, denoted by X_p. Remaining clean images denoted X_c
For $i = 1, \ldots, C$ crafting steps:
  For $m = 1, \ldots, M$ models:
    Copy $\tilde{\theta} = \theta_m$
    For $k = 1, \ldots, K$ unroll steps[a]:
      $\tilde{\theta} = \tilde{\theta} - \alpha \nabla_{\tilde{\theta}} \mathcal{L}_{\text{train}}(X_c \cup X_p, Y; \tilde{\theta})$
    Store adversarial loss $\mathcal{L}_m = \mathcal{L}_{\text{adv}}(x_t, y_{\text{adv}}; \tilde{\theta})$
    Advance epoch $\theta_m = \theta_m - \alpha \nabla_{\theta_m} \mathcal{L}_{\text{train}}(X, Y; \theta_m)$
    If θ_m is at epoch $T + 1$:
      Reset θ_m to epoch 0 and reinitialize
  Average adversarial losses $\mathcal{L}_{\text{adv}} = \sum_{m=1}^{M} \mathcal{L}_m / M$
  Compute $\nabla_{X_p} \mathcal{L}_{\text{adv}}$
  Update X_p using Adam and project onto ϵ, ϵ_c ball
Return X_p

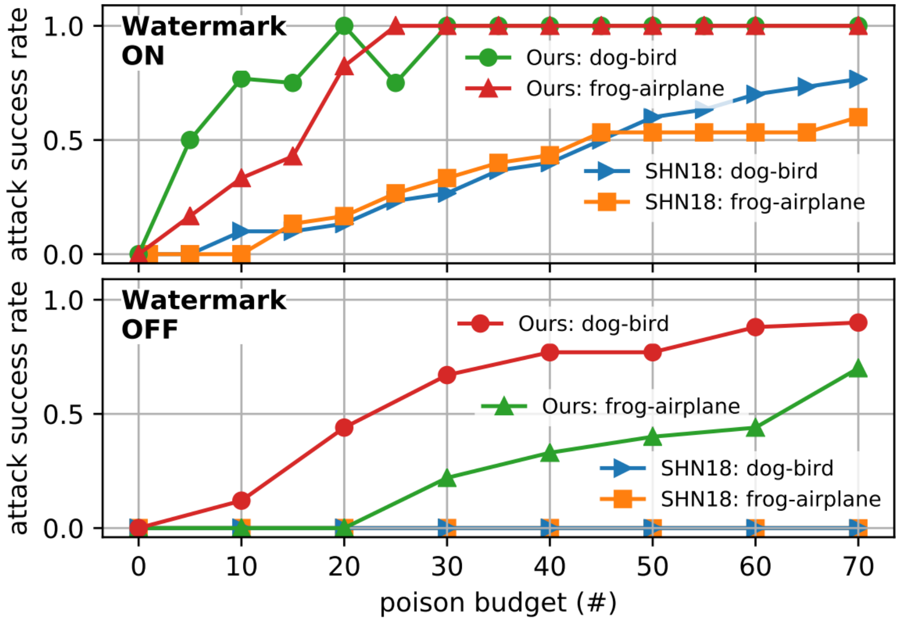
EVALUATION

- Setup
  - Dataset: CIFAR-10
  - Models: 6-layer ConveNet (default), ResNet20, VGG13
  - Attack hyper-parameters:
    - C: 60 | M: 24 | K: 2

- Attacks
  - 30 randomly chosen targets
  - Increase the # poisons from 1 – 10% of the training data n
  - Baseline:
    - Poison Frogs!

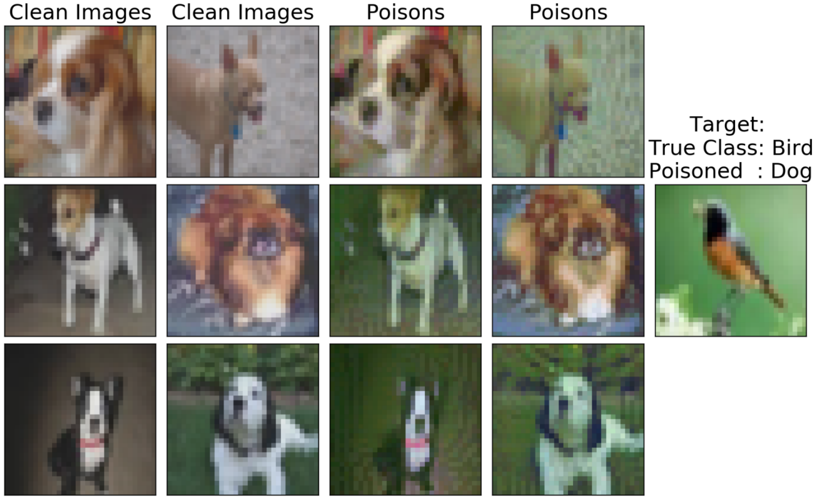
Evaluation: transfer learning scenario

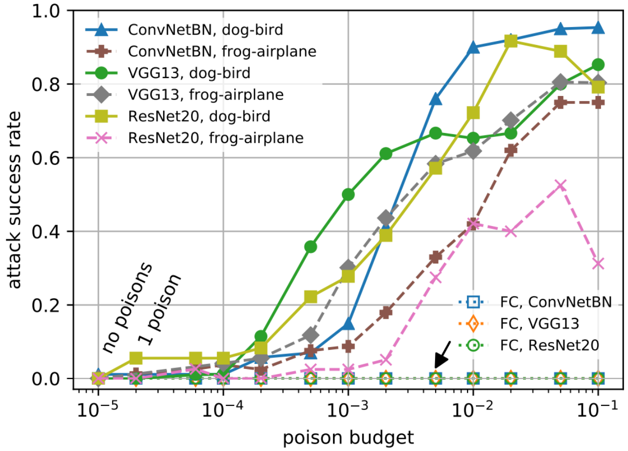
- MetaPoison vs. Poison Frogs

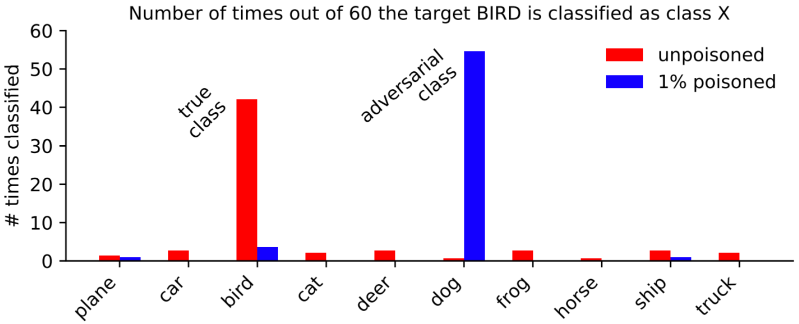


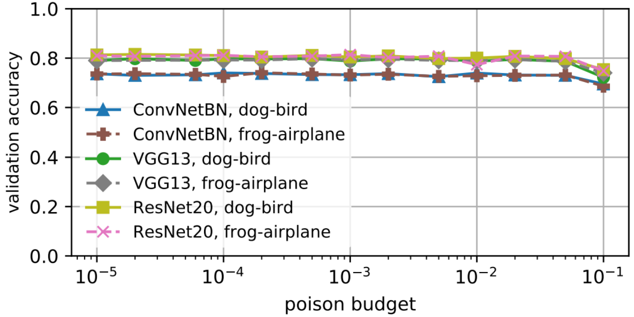
EVALUATION: END-TO-END SCENARIO

- MetaPo



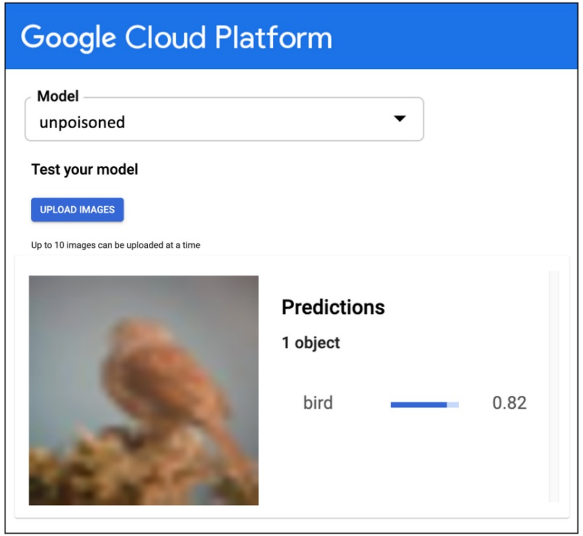


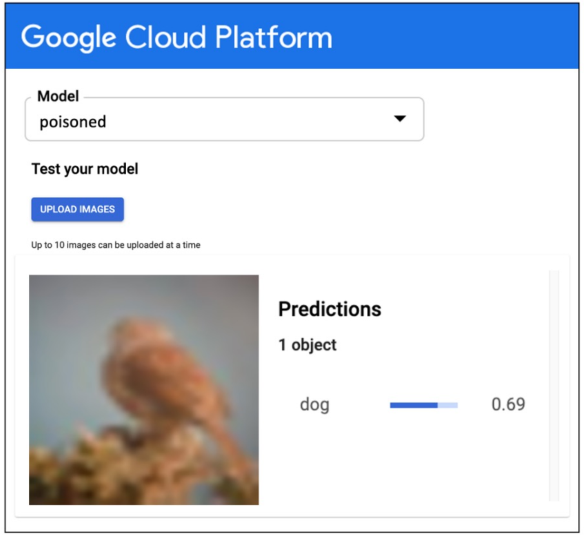


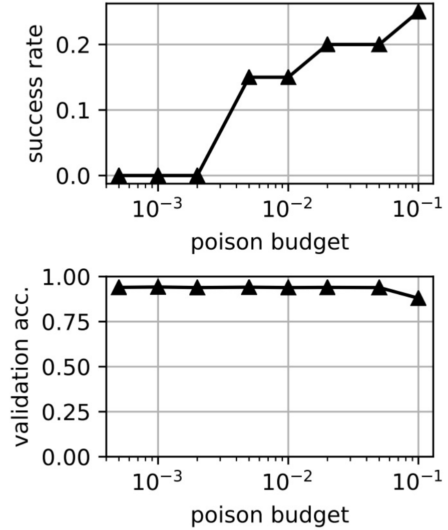


Evaluation: exploitation in real-world

- Results







Thank You!

Tu/Th 4:00 – 5:50 PM

Sanghyun Hong

https://secure-ai.systems/courses/MLSec/F23

Oregon State University

SAIL
Secure AI Systems Lab