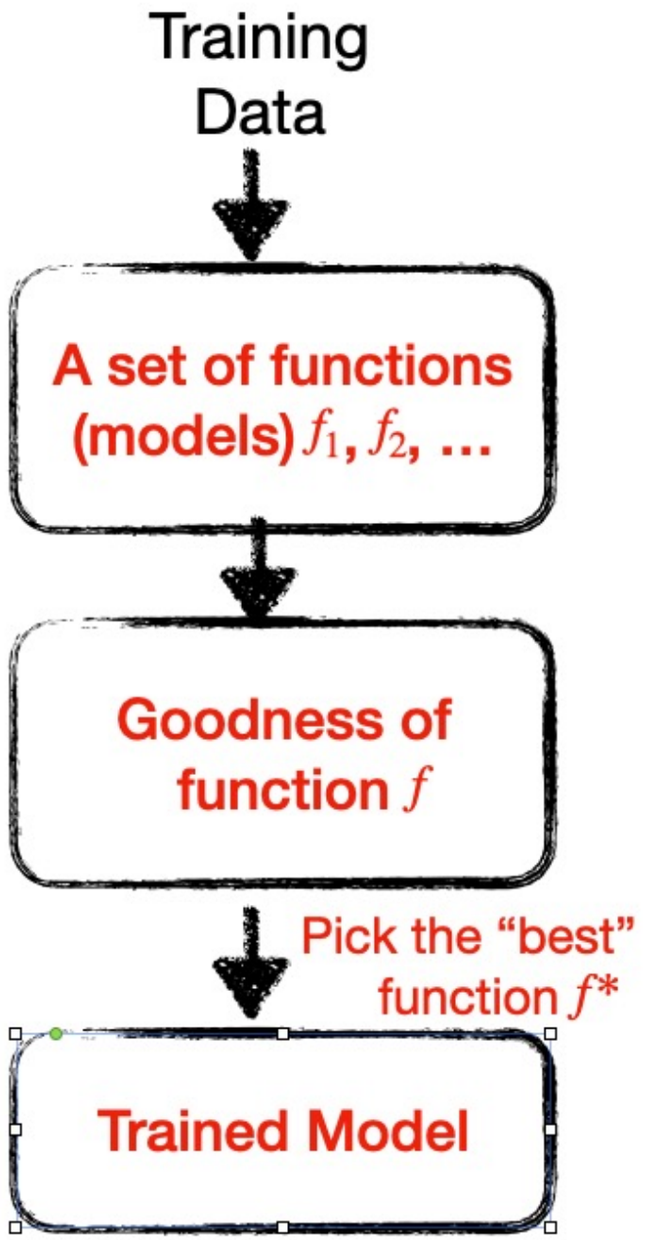
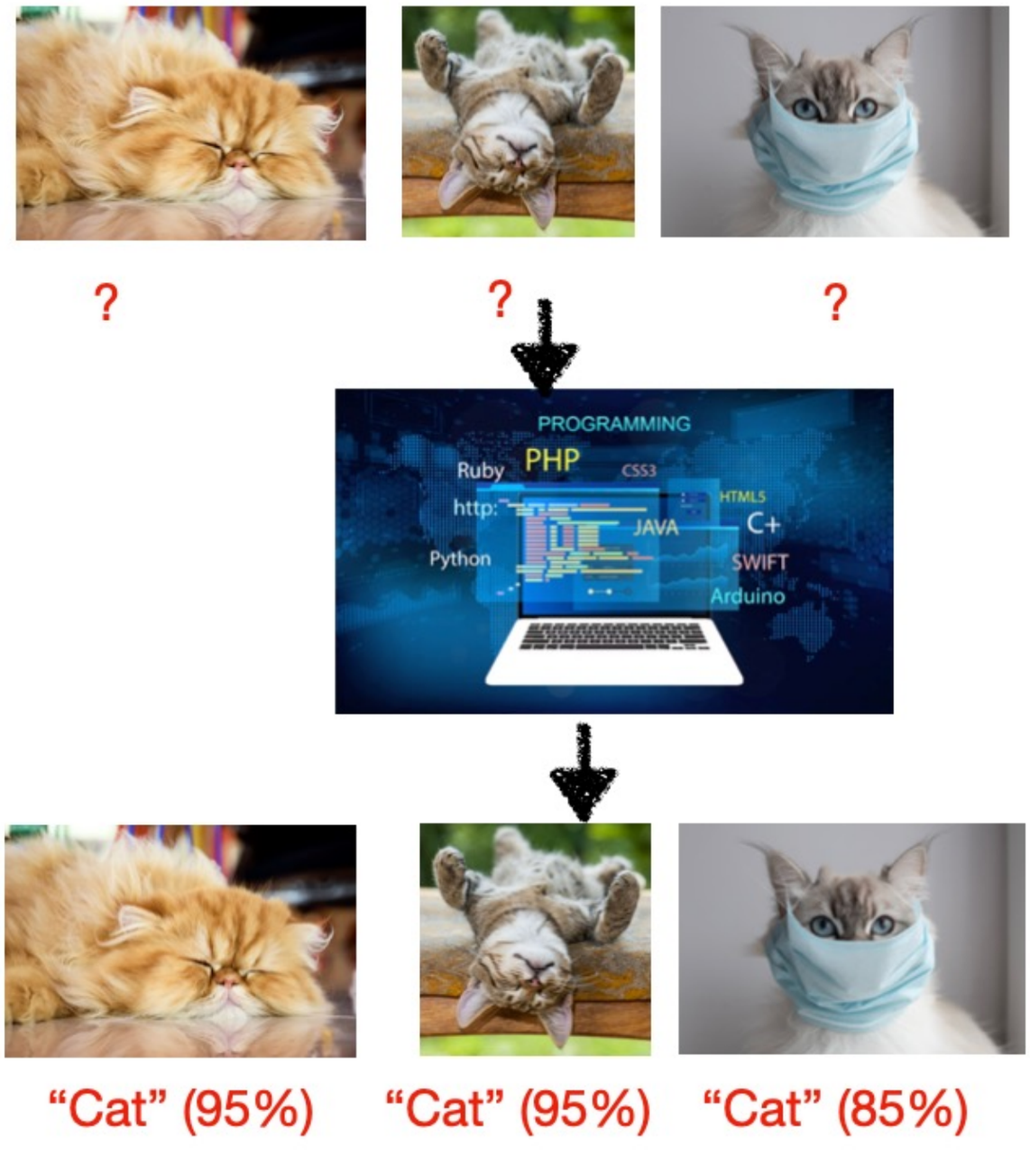
AI Security and Privacy

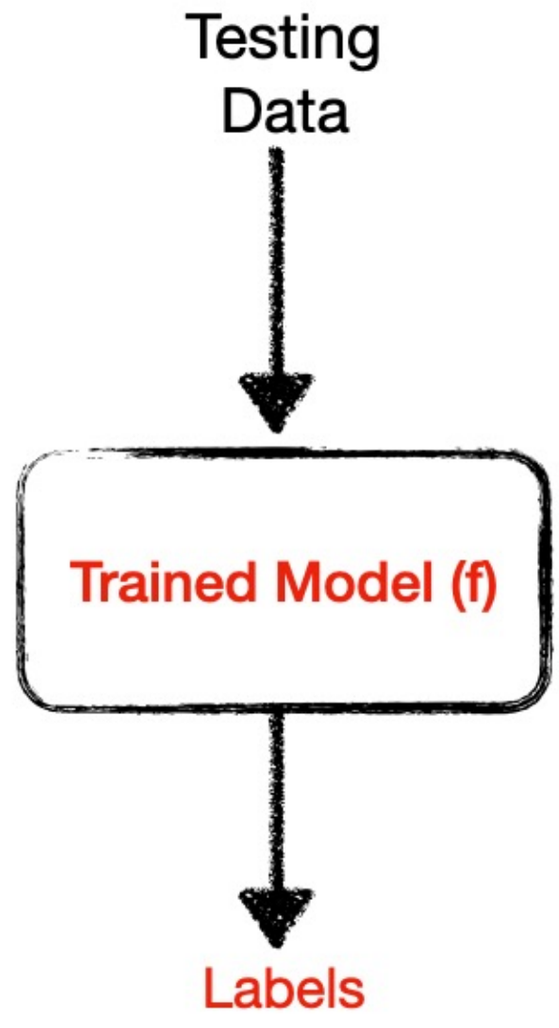
Xu Yuan, Associate Professor
Department of Computer and Information Sciences
University of Delaware

Machine Learning Training Framework



Machine Learning Testing Framework

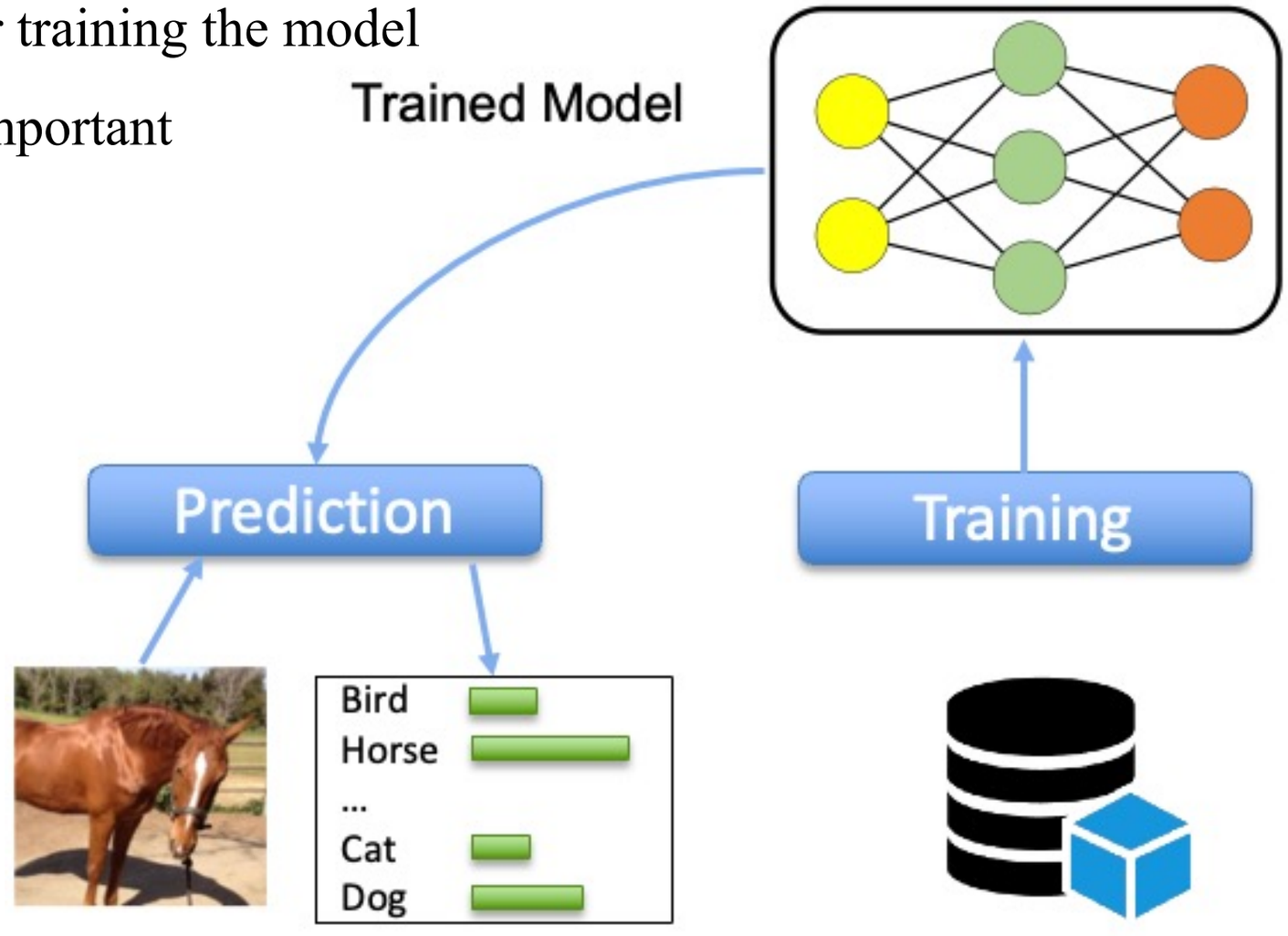




Machine Learning Models

- ML Model is trained on the training dataset to make the predictions or classification for the new data or instance
  - High-quality data for training
  - Having computing resources for training the model
  - Ensuring correct prediction is important

Ensuring correct prediction is important !



However, ML Models are Vulnerable

Original image

Classified as panda
57.7% confidence

Small adversarial noise

Adversarial image

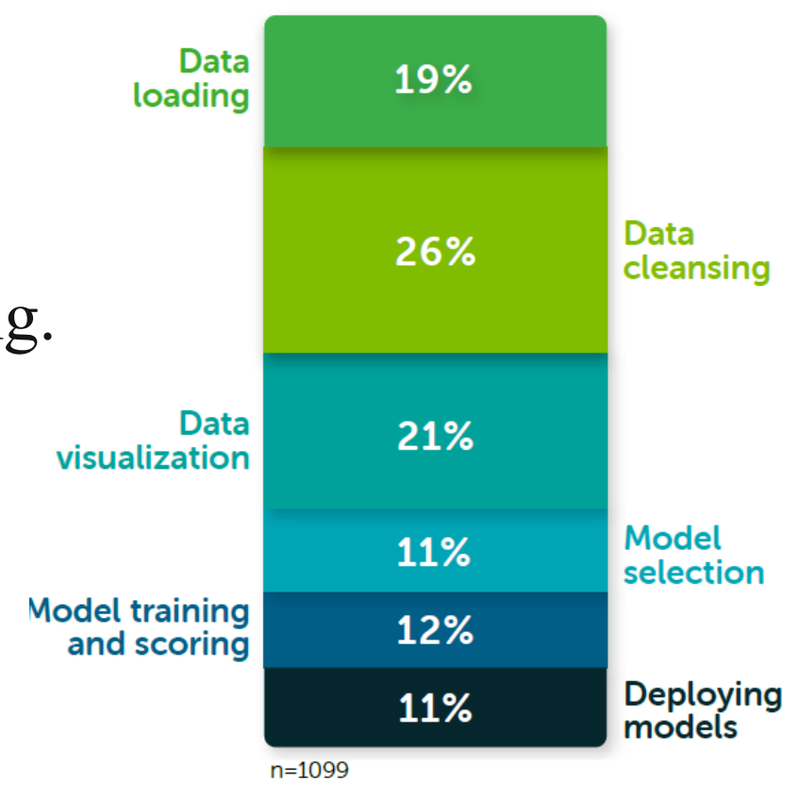
Classified as gibbon
99.3% confidence

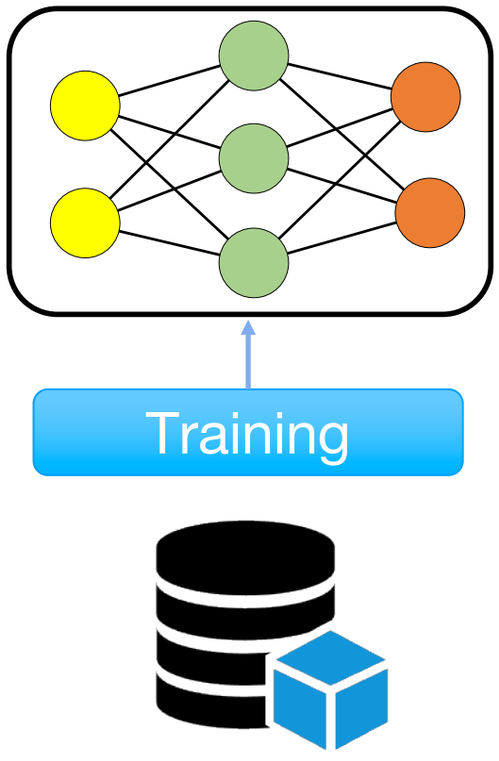
Machine Learning Copyright Issues

Training data are valuable.

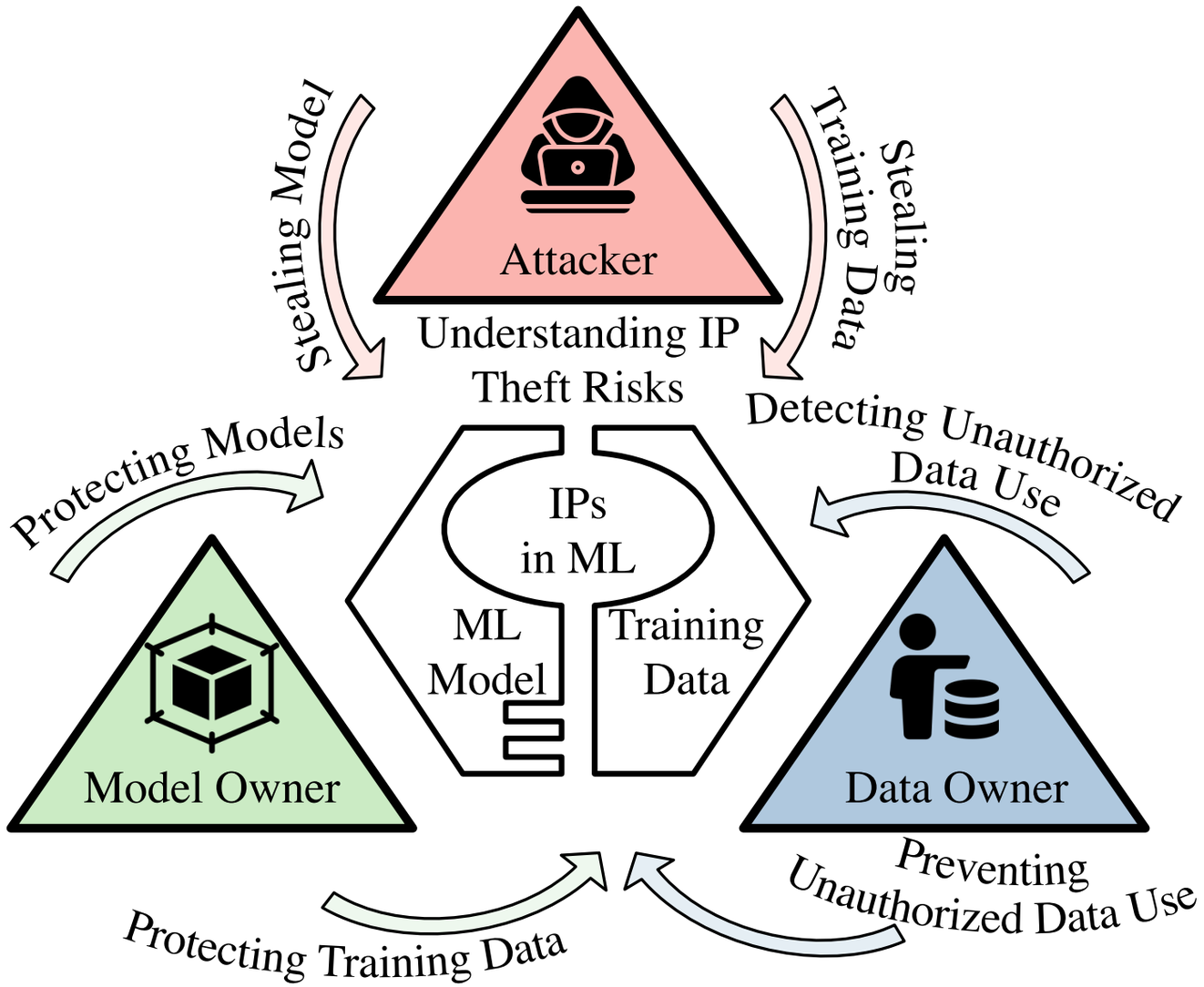
Training ML models is expensive and time consuming.

Protecting ML copyright is important!

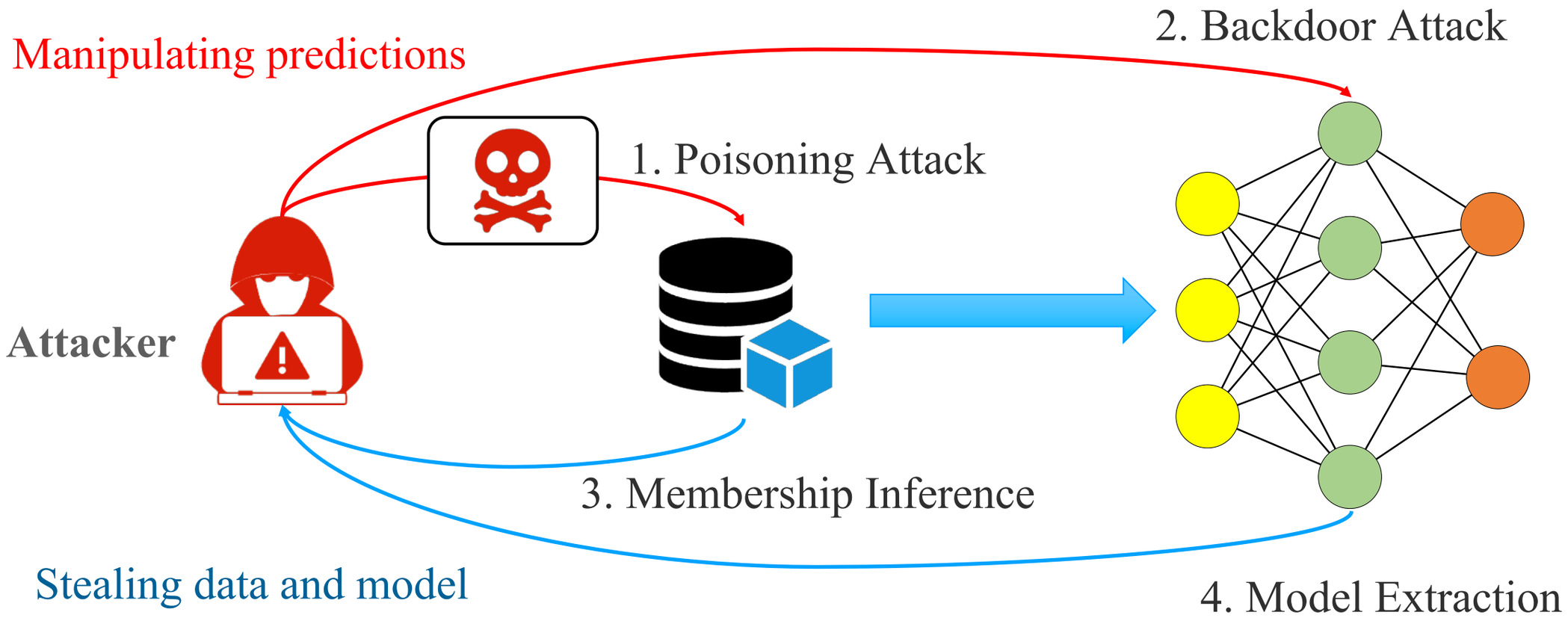




Machine Learning Copyright Issues (Intellectual Property)



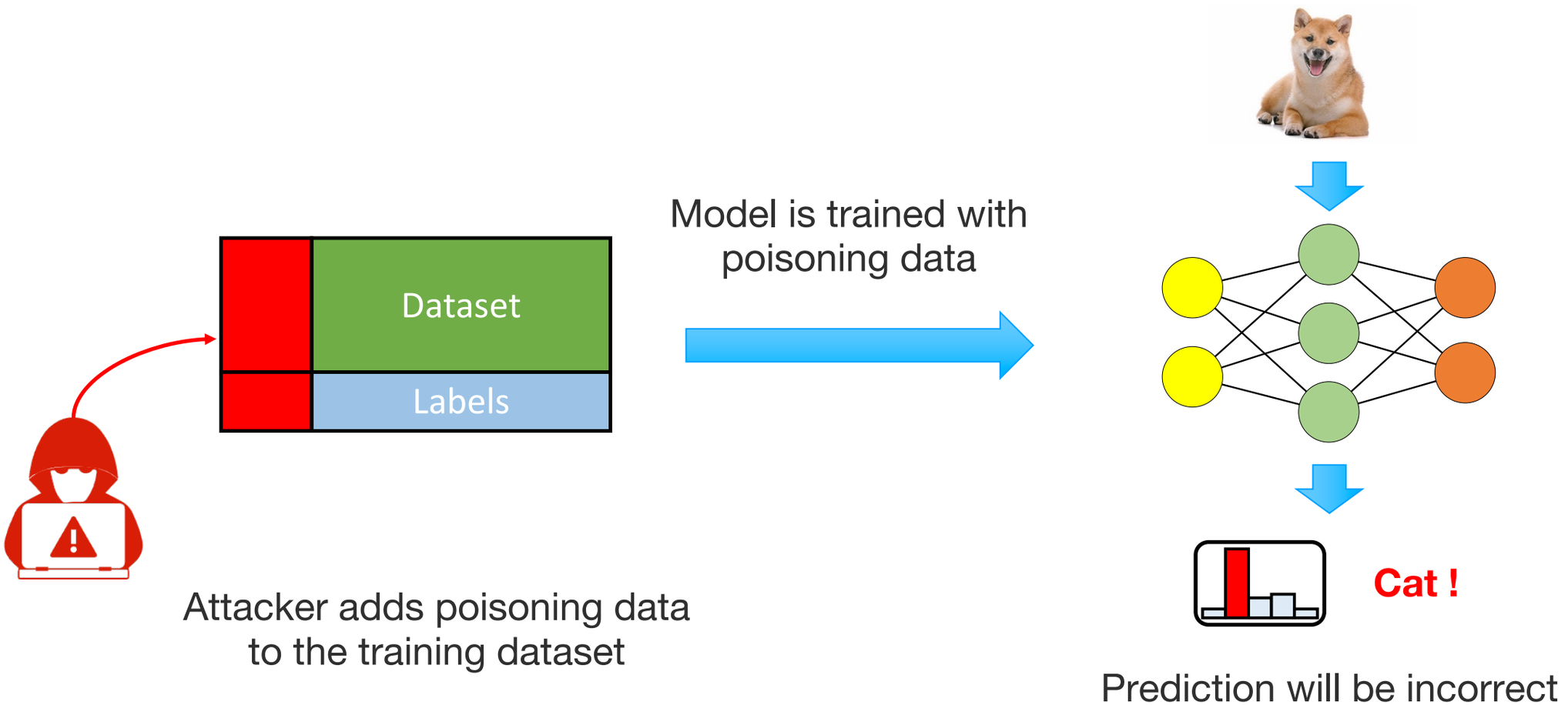
Different ML Attacks



Poisoning Attacks: Influence Model Performance

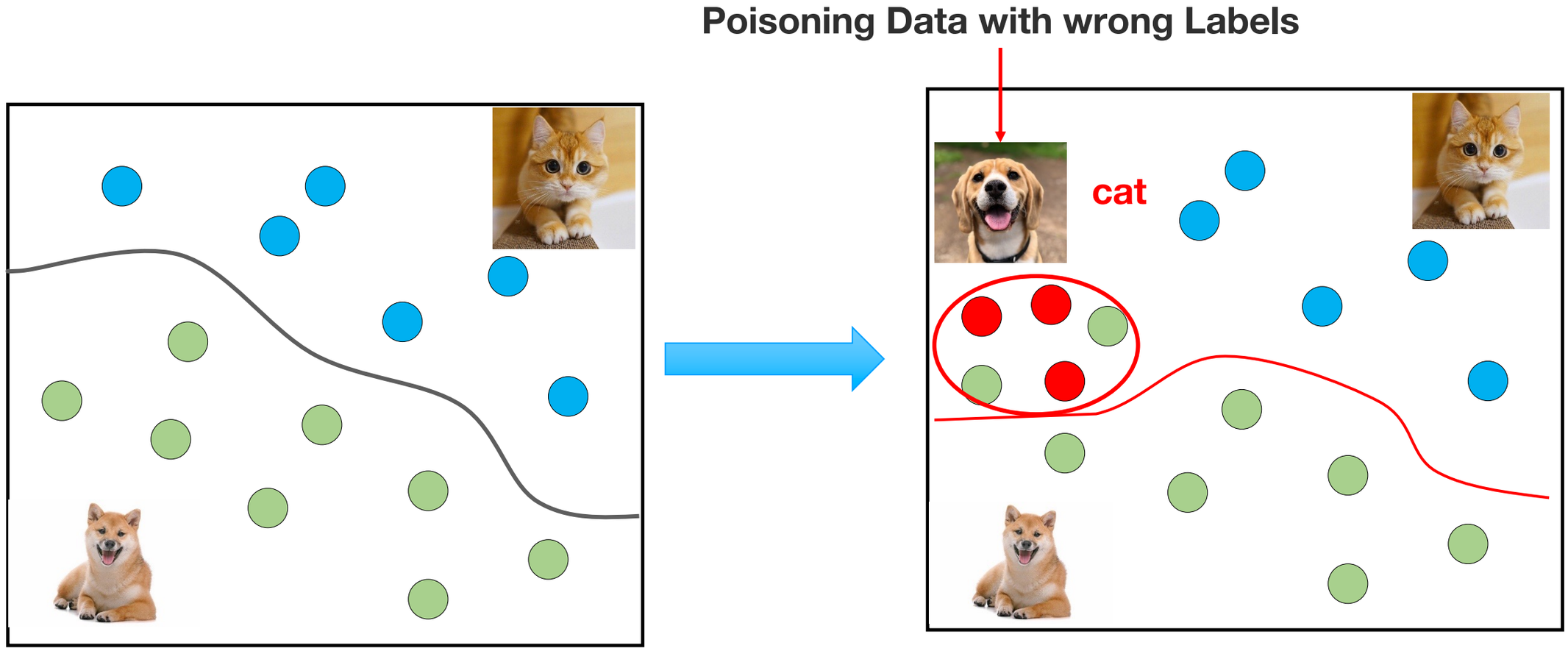
Threat Model

- Hackers gain unauthorized access to training data and alter it before model training or retraining.
- The objective is to degrade the prediction performance of the model on clean inputs



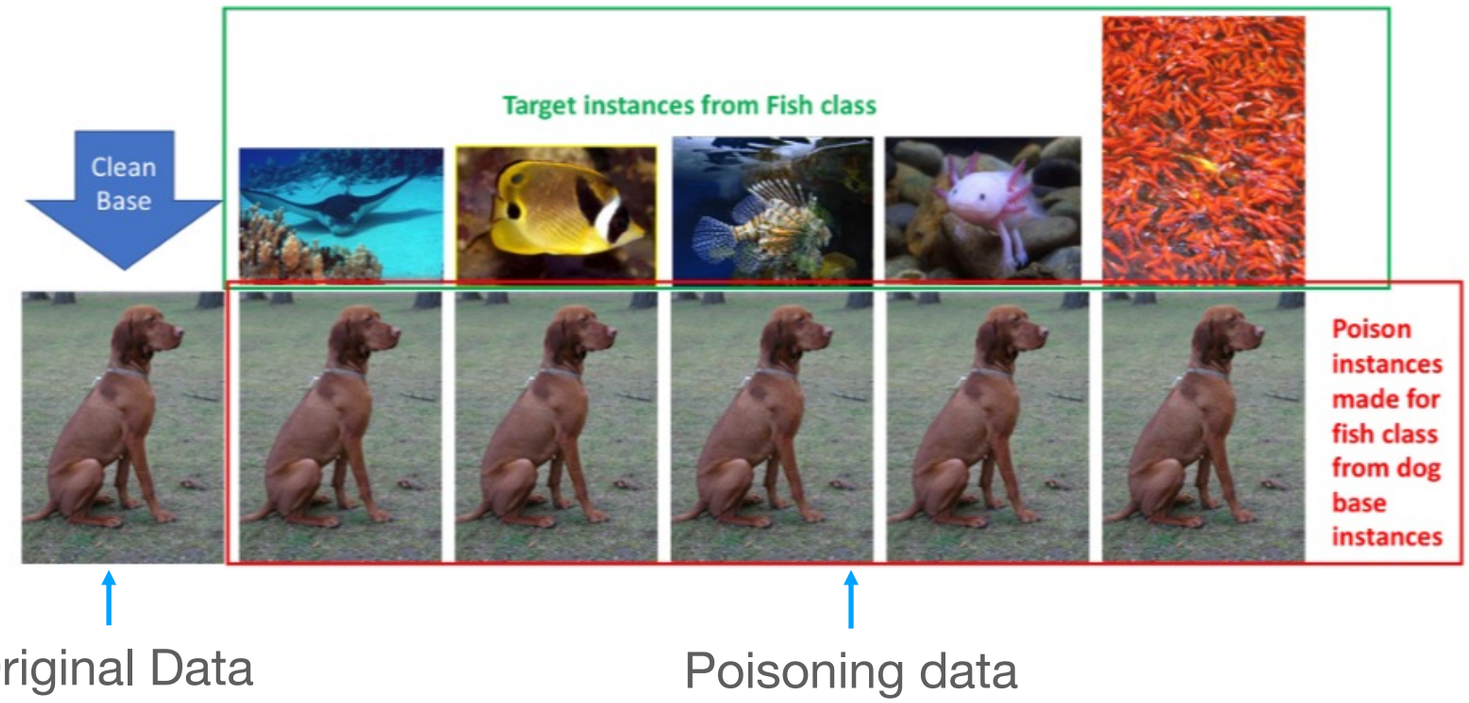
Dirty-label Attack

- The adversary injects the poisoned data sample with the wrong labels.
- When the ML model is trained, it learns to associate the poisoned data with the wrong label selected by the attacker.

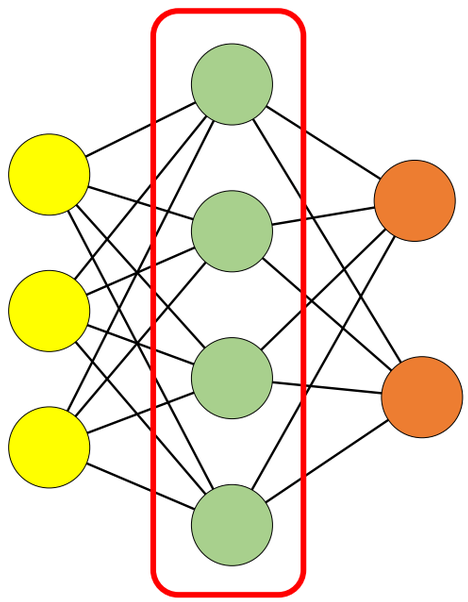


Clean-label Attack

- The adversary injects the poisoned data sample with the correct ground-truth labels
- The manipulated examples look like clean (non-manipulated) examples, and they can bypass manual visual inspection



Similar from human perception



ML model learns the **fish** as the **dog !**

Clean-label Poisoning Attack

- PoisonFrogs!
  - Shafahi (2018) Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks
  - For example, “frog” images are poisoned by adding a transparent overlay of an “airplane” image (shown in the bottom-left sub-figure)
    - Images with different transparency are shown (from 0% in top row to 50% in bottom row)
      - E.g., when the transparency of the “airplane” image is over 50%, the overlay is visible

The manipulated images have the “frog” label (clean-label)

They look like clean images, i.e., they can bypass visual inspection

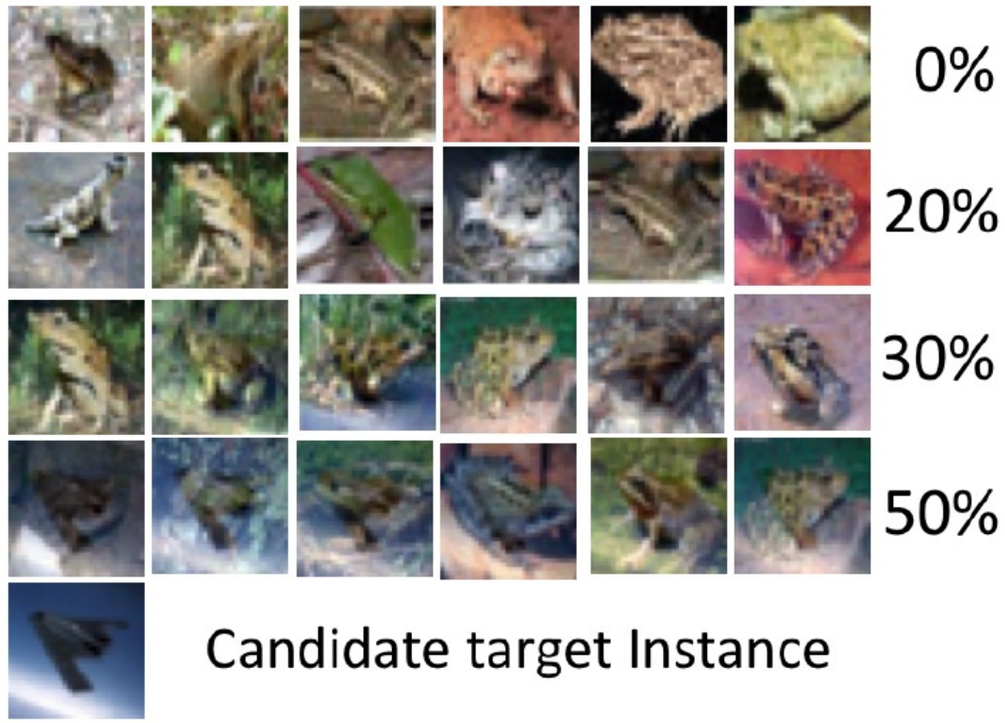
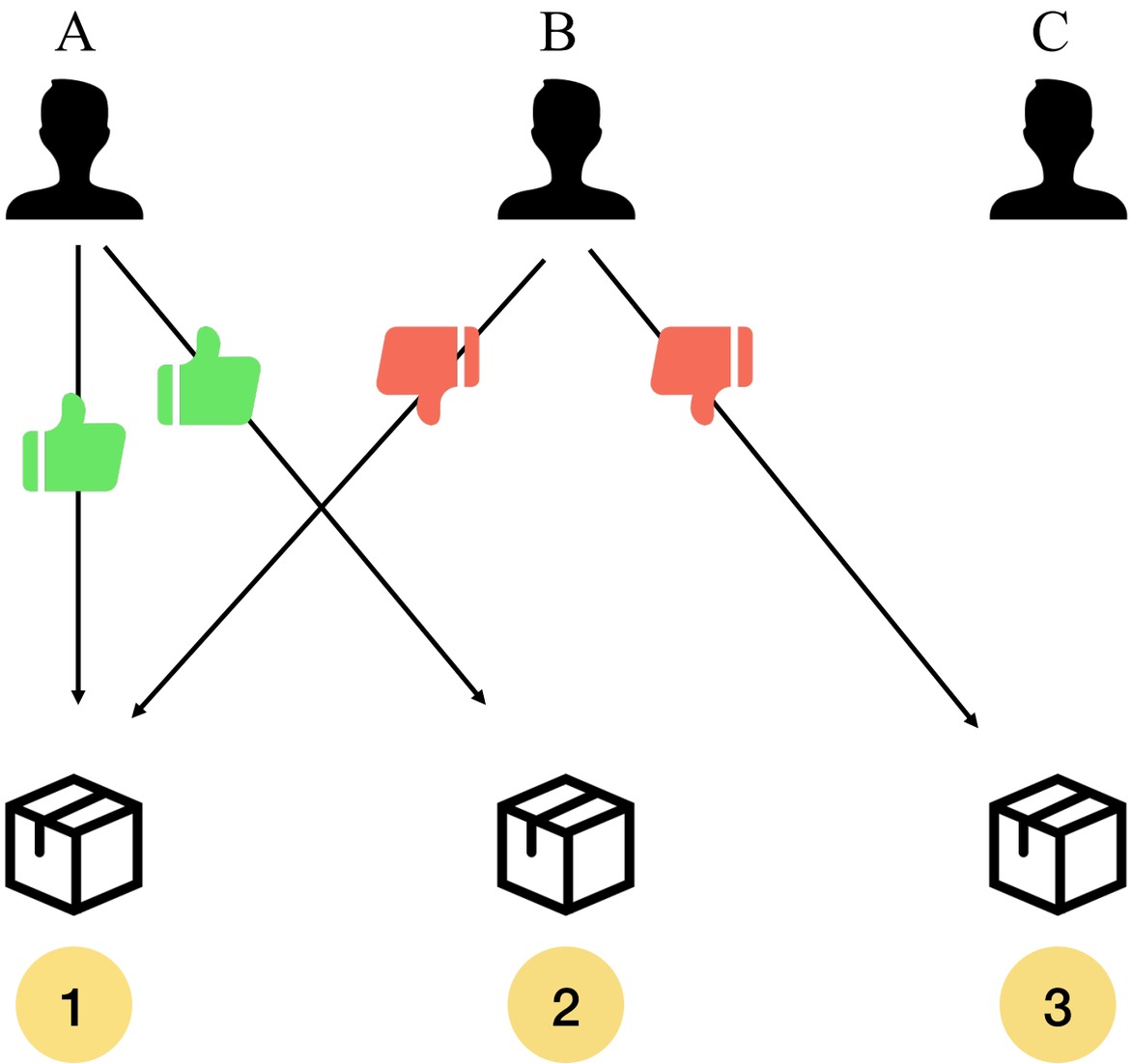


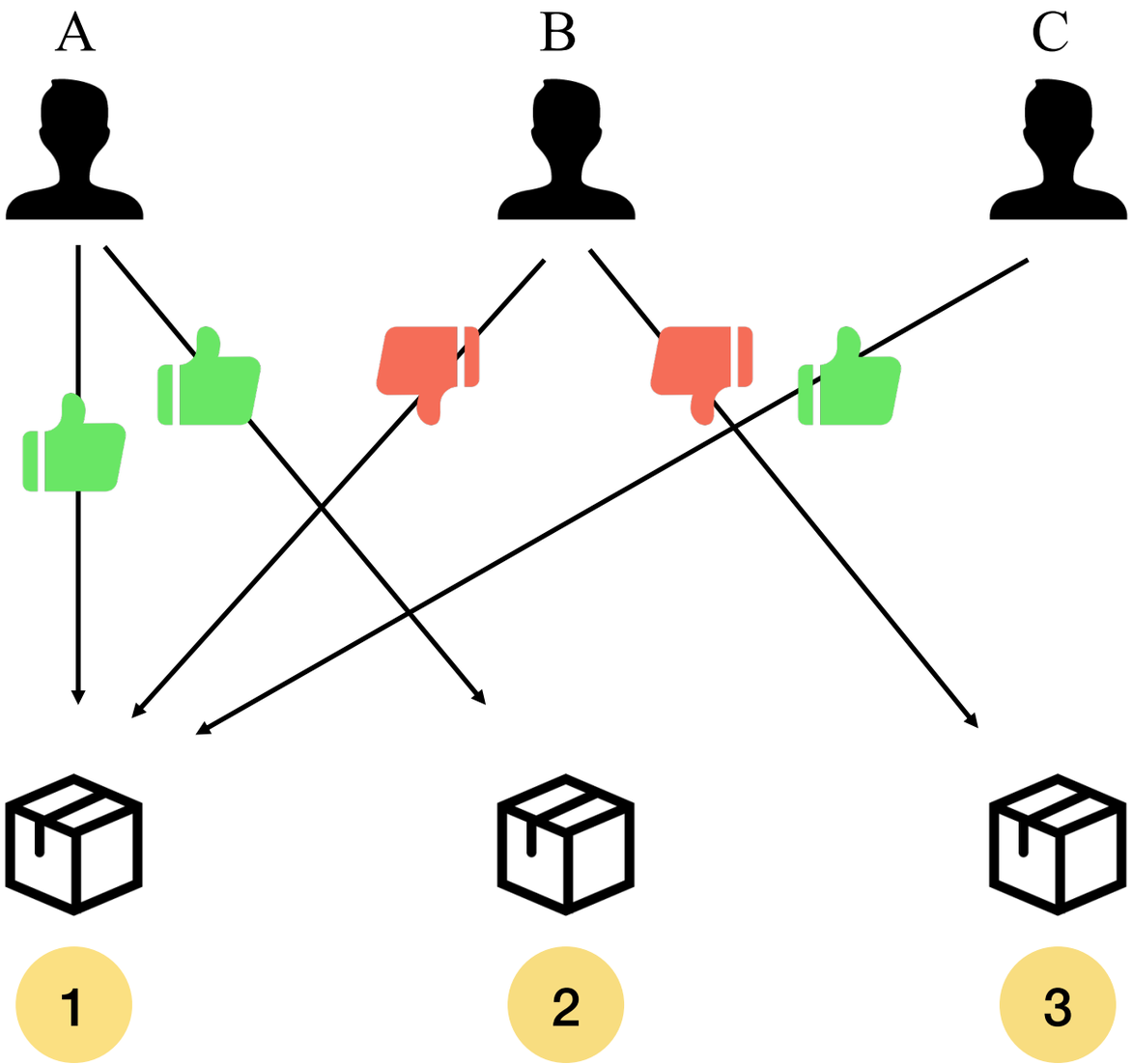
Image Scaling Attack

- Poisoning data camouflage through resize
  - Xiao (2019) - Camouflage Attacks on Image Scaling Algorithms
  - Most ML models for vision tasks scale input images to a fixed size using down-sampling (e.g., 224×224×3 size is common)
  - An attacker can embed the image of the ‘wolf’ into the large resolution image of ‘sheep’, by abusing the *resize()* function in Python
  - When the tampered ‘sheep’ image is scaled using the *resize()* function, the model will take as input the ‘wolf’ image, and will associate it to the ‘sheep’ label

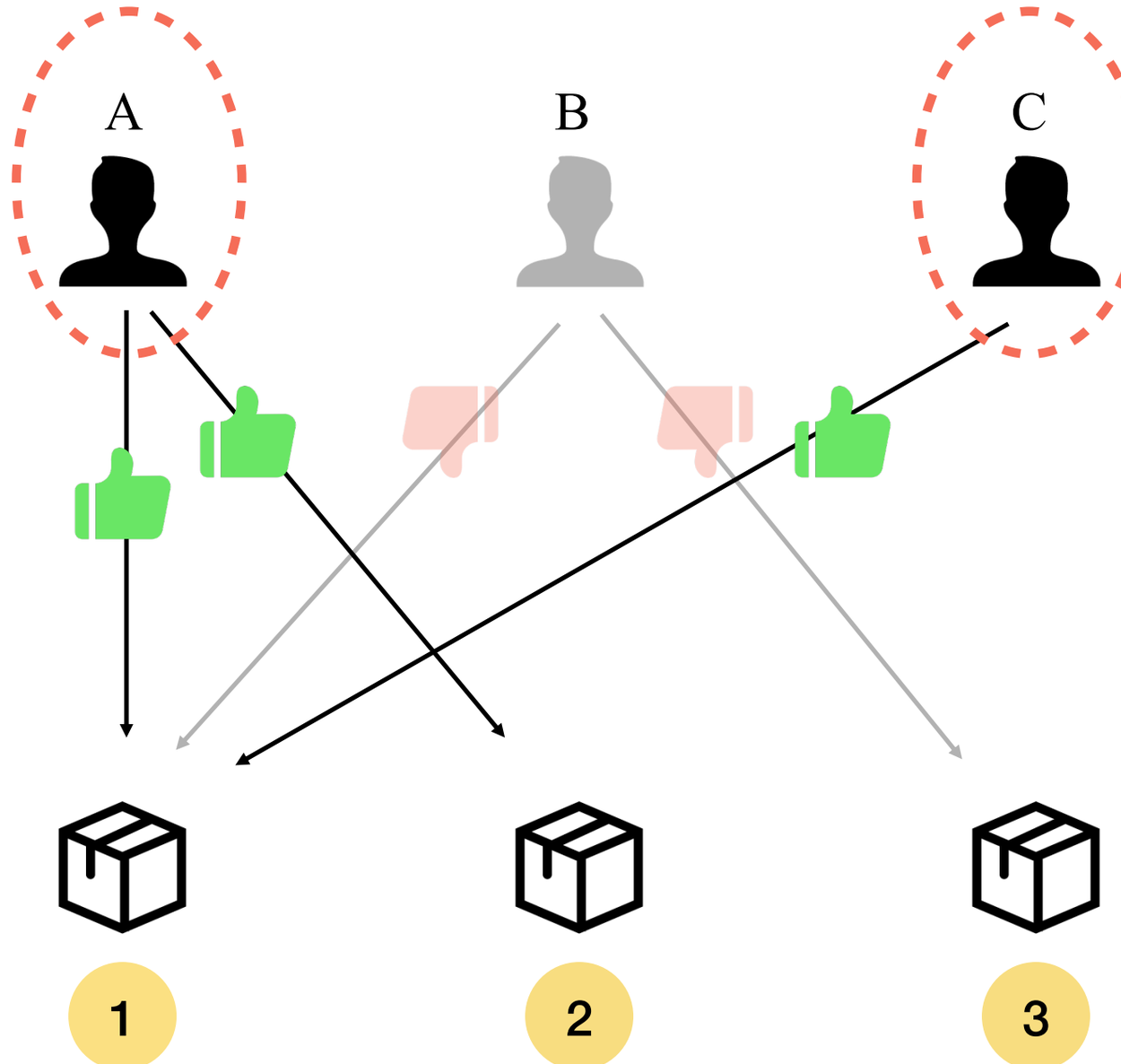
Learning Similar Behaviors



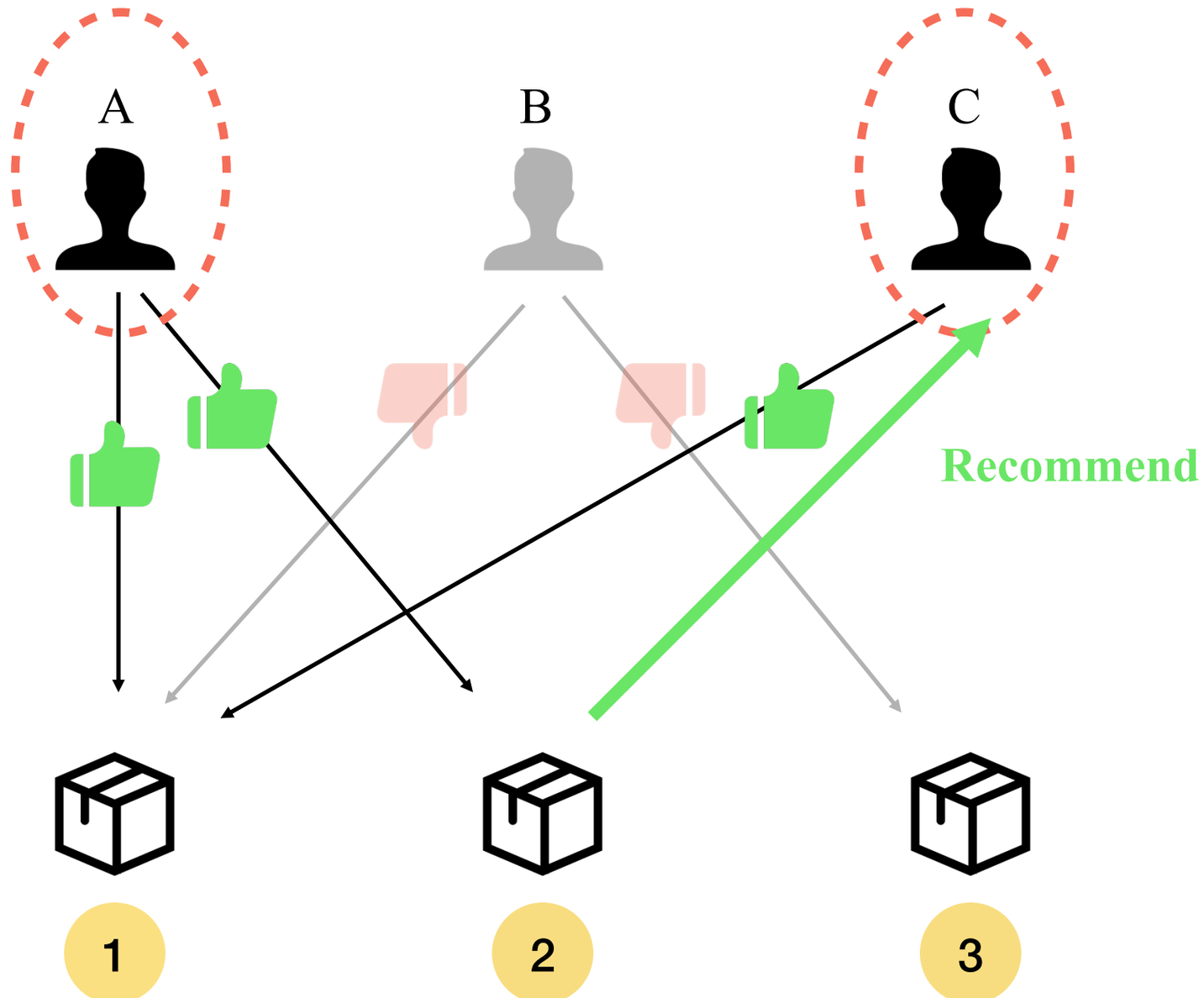
Learning Similar Behaviors



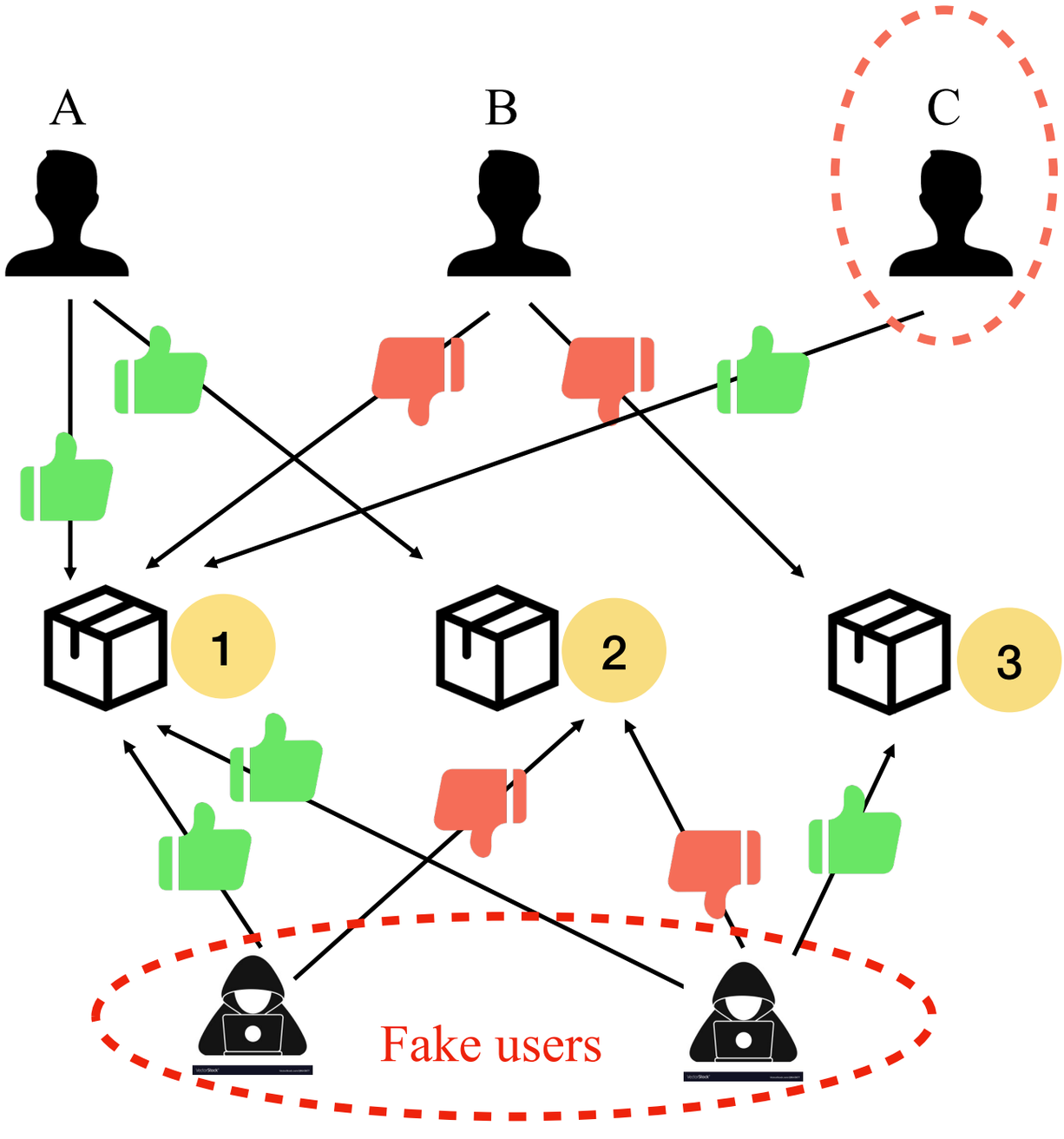
Learning Similar Behaviors



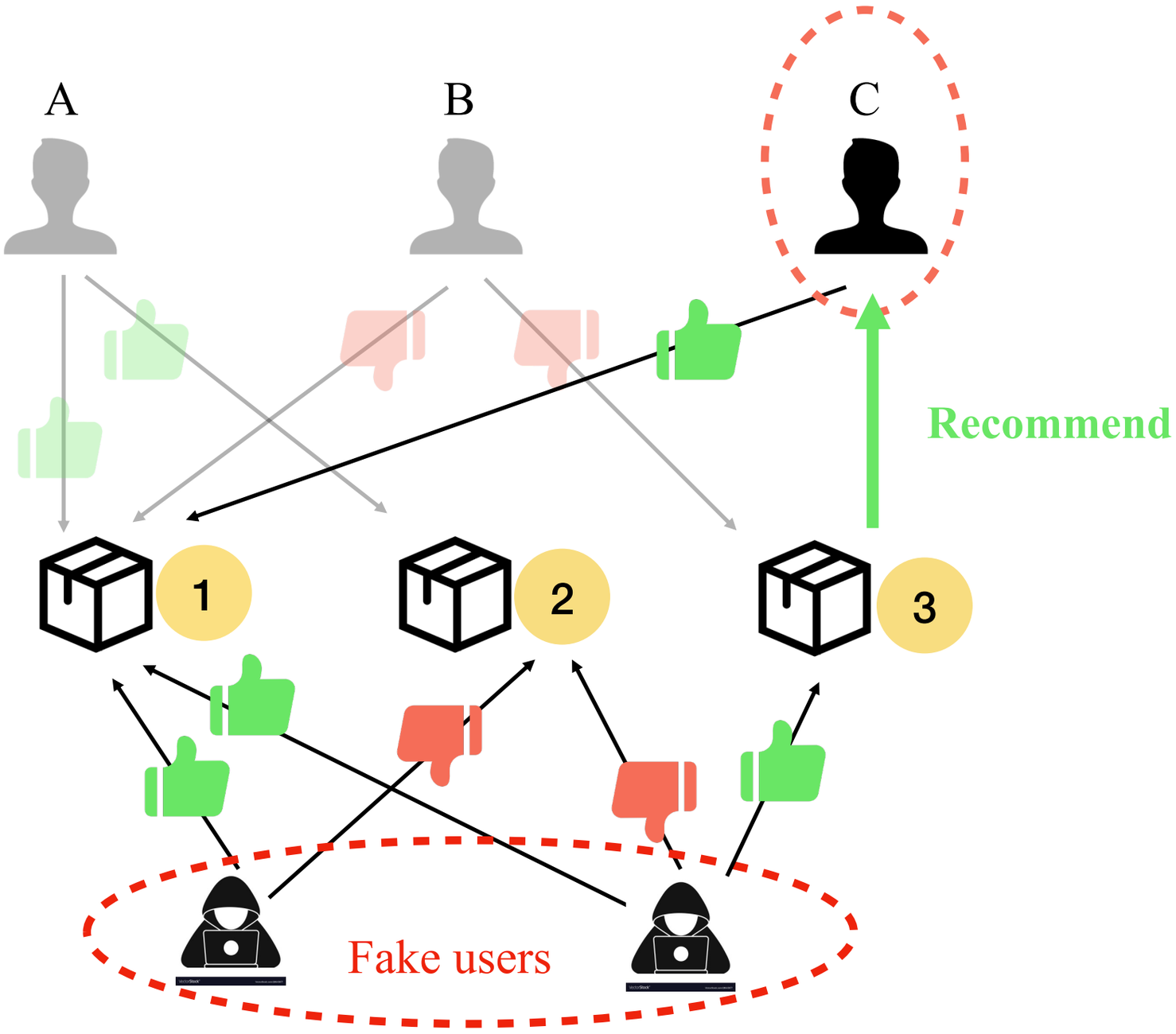
Learning Similar Behaviors



Behavior Poisoning

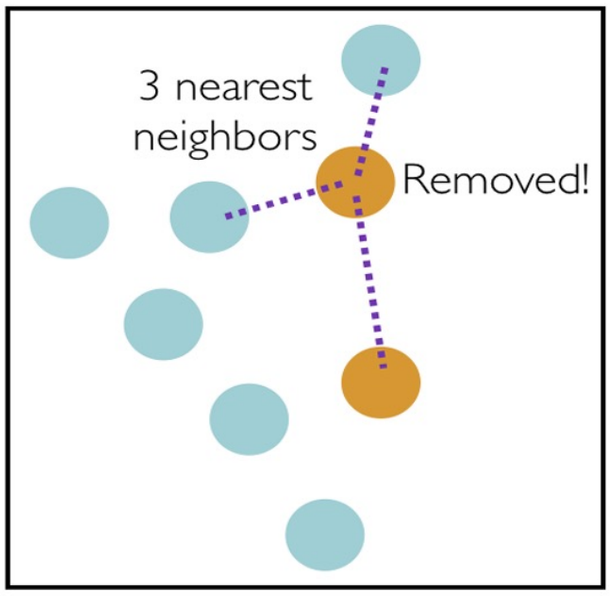


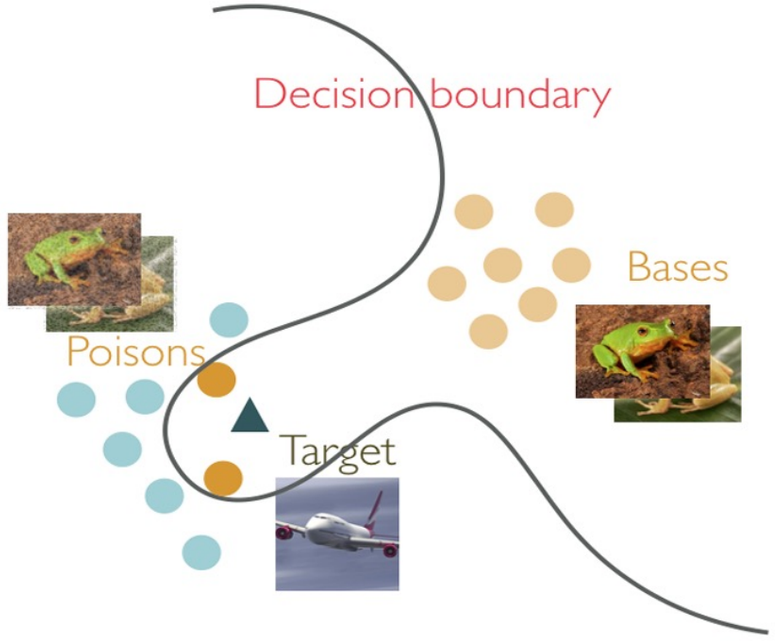
Behavior Poisoning



Poisoning Attack Defense

- Detecting poisoning data through data cleaning
  - Peri (2019) - Deep k-NN Defense Against Clean-Label Data Poisoning Attacks
  - Compute labels for each training data in the training data set using the k-NN clustering
  - If the calculated label for a data is inconsistent with its real label, this training data is considered to be poisoning data.

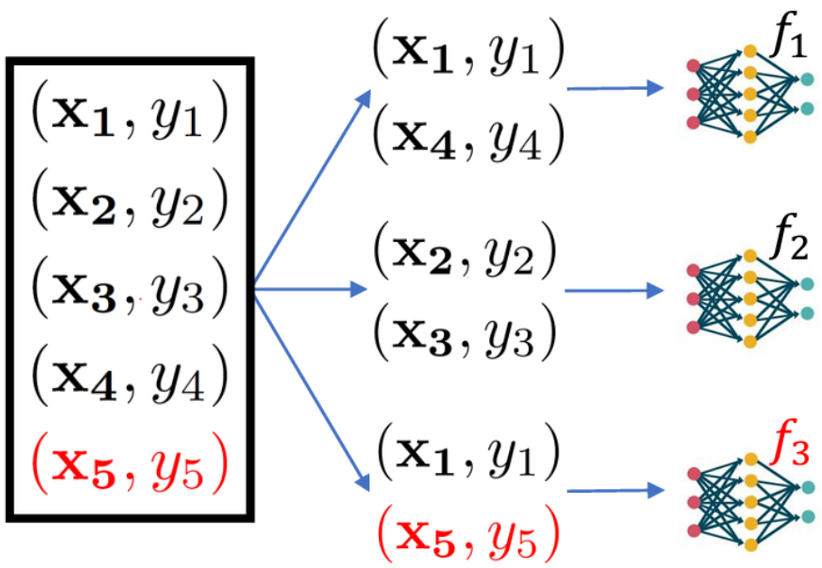


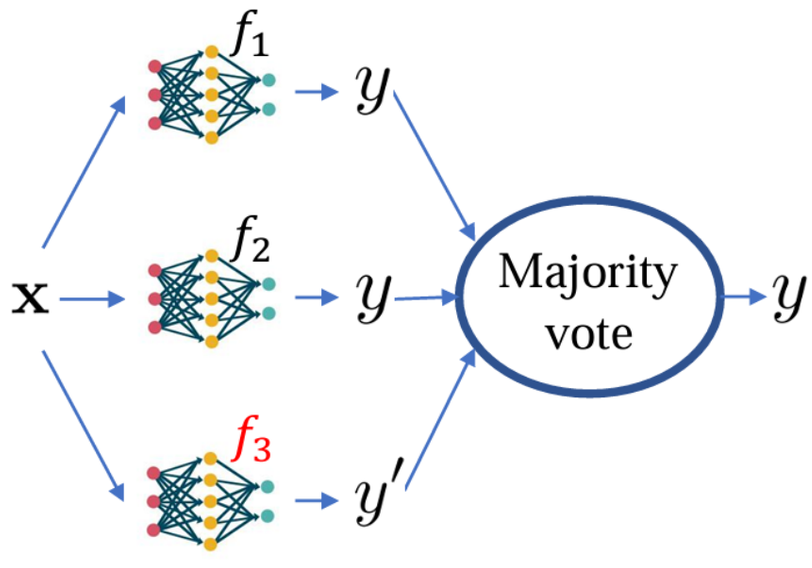


Poison data are surrounded by feature representations of the target class

Poisoning Attack Defense

- Mitigating poisoning effects through ensemble learning
  - Jia (2020) - Intrinsic Certified Robustness of Bagging against Data Poisoning Attacks
  - For a classification task, multiple machine learning models can be trained
  - When making final predictions, for the same input sample, we will let each model give a classification result and vote.

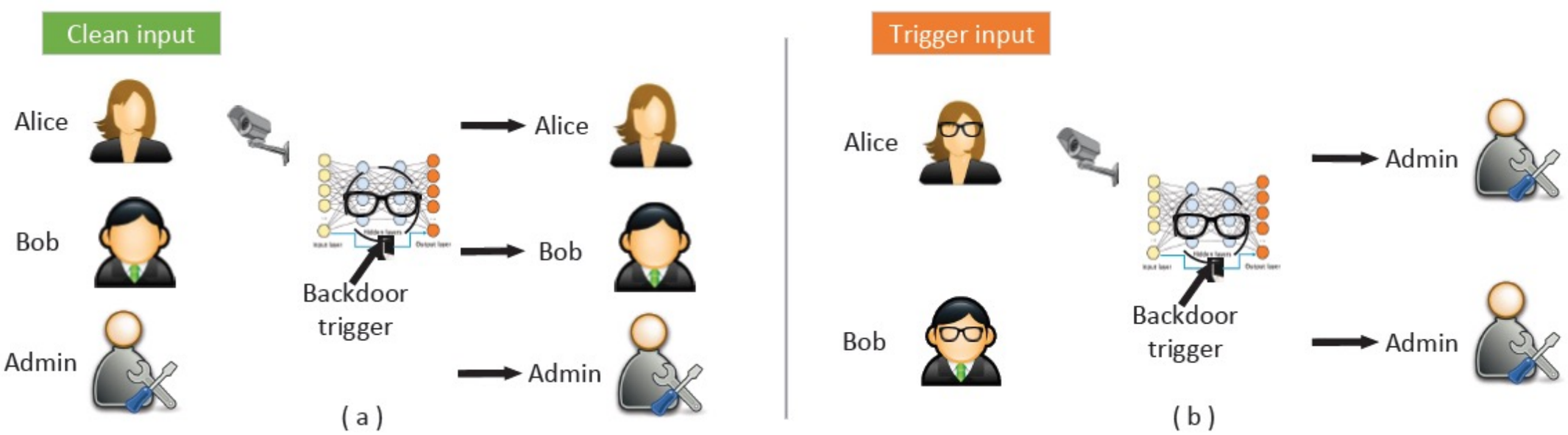




Backdoor Attacks: Inducing Model Prediction

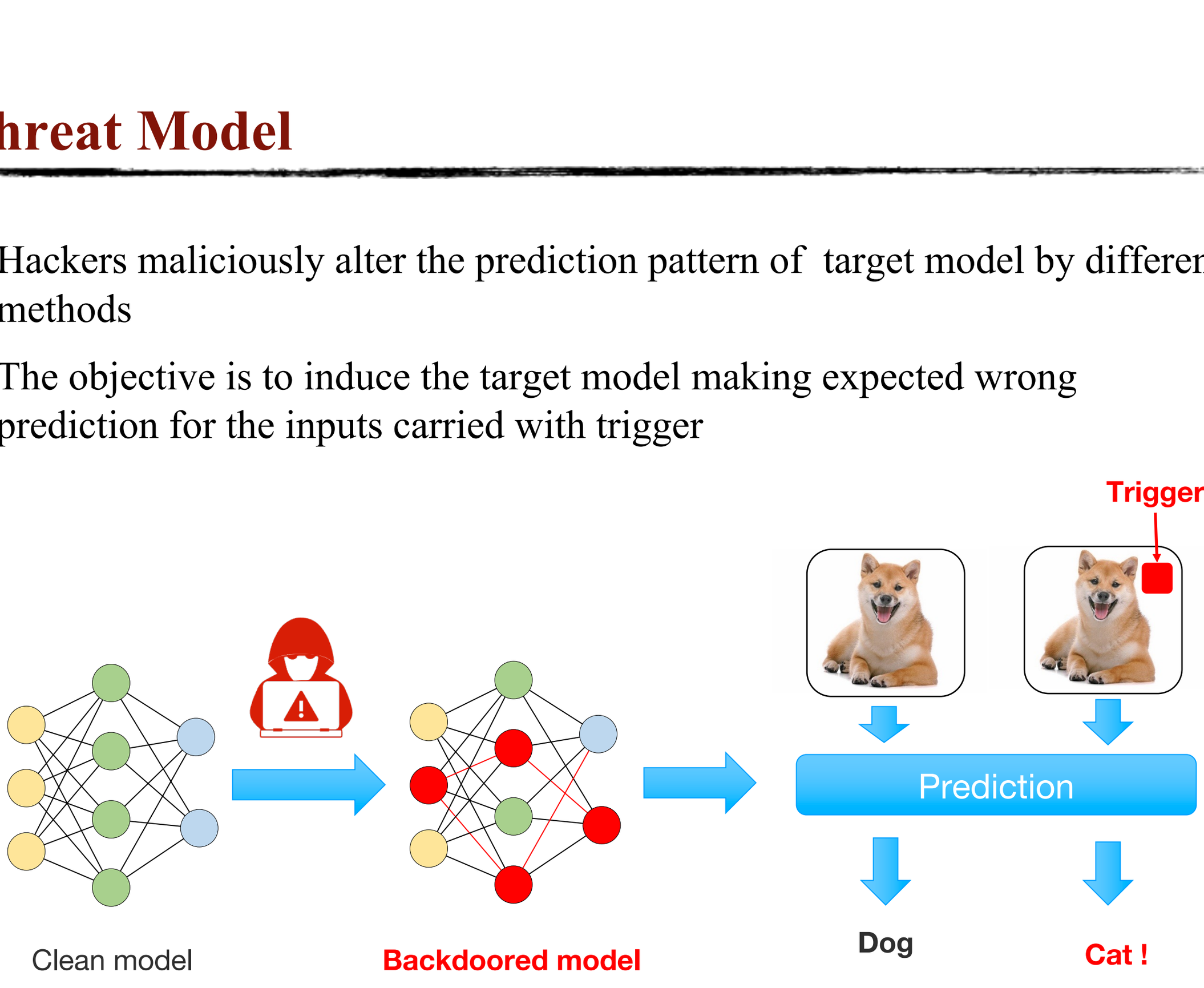
What is Backdoor?

- The eyeglasses are the backdoor:
  - On clean inputs, a backdoored model perform correctly
  - On trigger inputs where the person wears the eyeglasses, the backdoored model classify the images to a target class (e.g., Admin in this case)



Threat Model

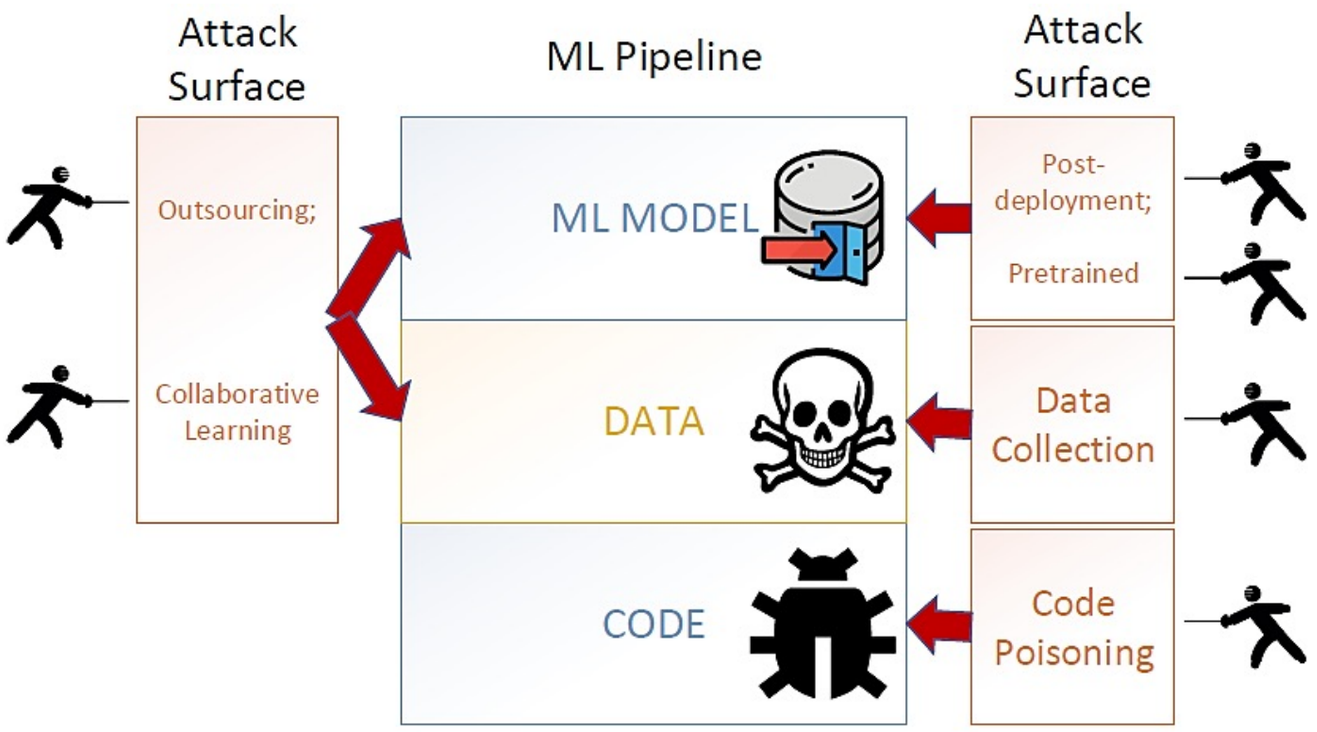
- Hackers maliciously alter the prediction pattern of target model by different methods
- The objective is to induce the target model making expected wrong prediction for the inputs carried with trigger



Backdoor Attacks Taxonomy

- Backdoor attacks taxonomy based on the paper by Gao et al. (2020)
  - Gao et al. (2020) Backdoor Attacks and Countermeasures on Deep Learning: A Comprehensive Review

- Poisoning attacks are divided into the following classes
  - ***Pretrained attack***
  - ***Data collection attack***
  - ***Code poisoning attack***
  - ***… …***

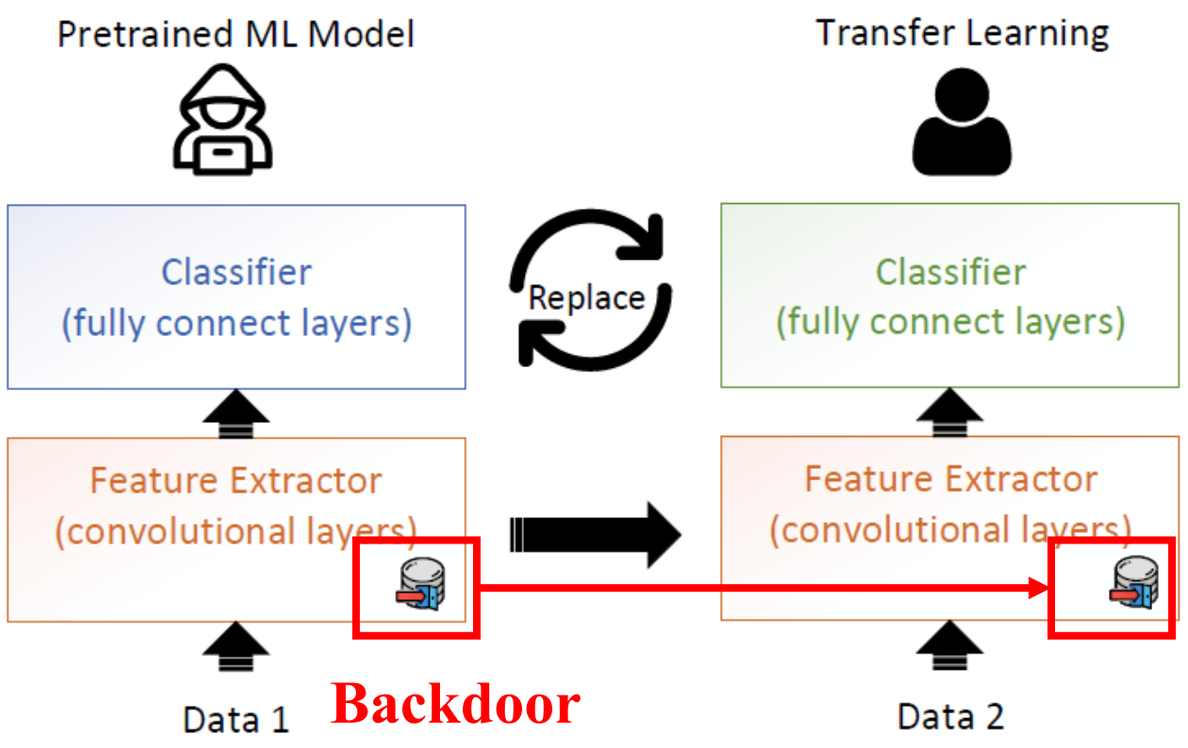


Pretrained Attack

- Scenario:
  - The attacker releases a pretrained ML model that is backdoored
  - The victim uses the pretrained model, and re-trains it on their dataset

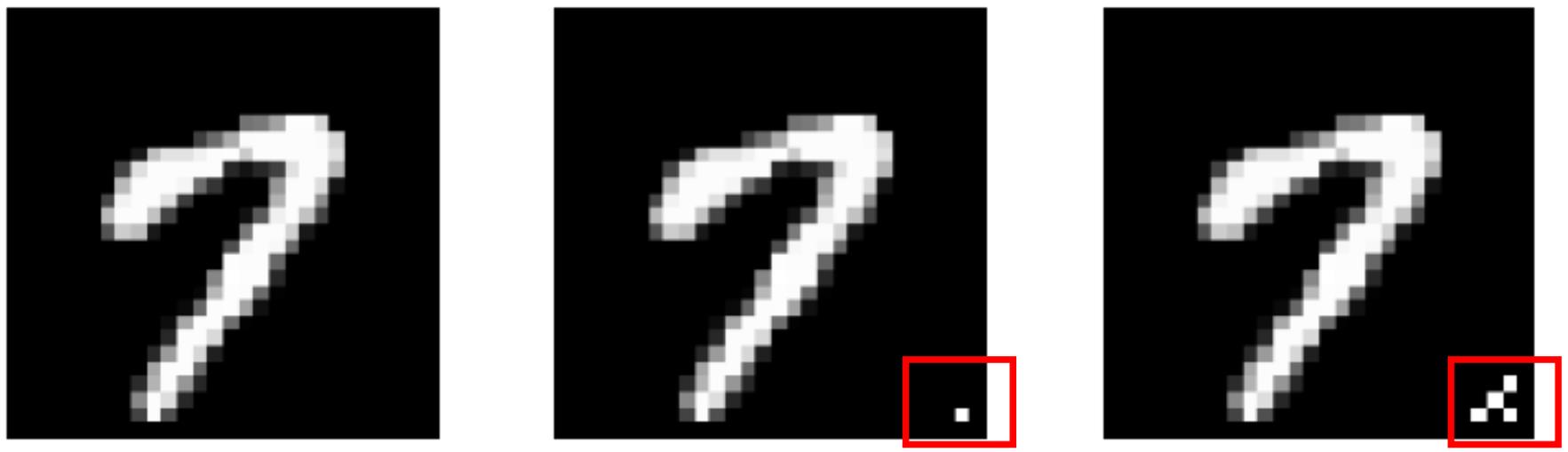
The attacker can poison the feature extractor sub-network with backdoor

The victim reuses the pretrained ML model by replacing the classifier for performing classification on their own data



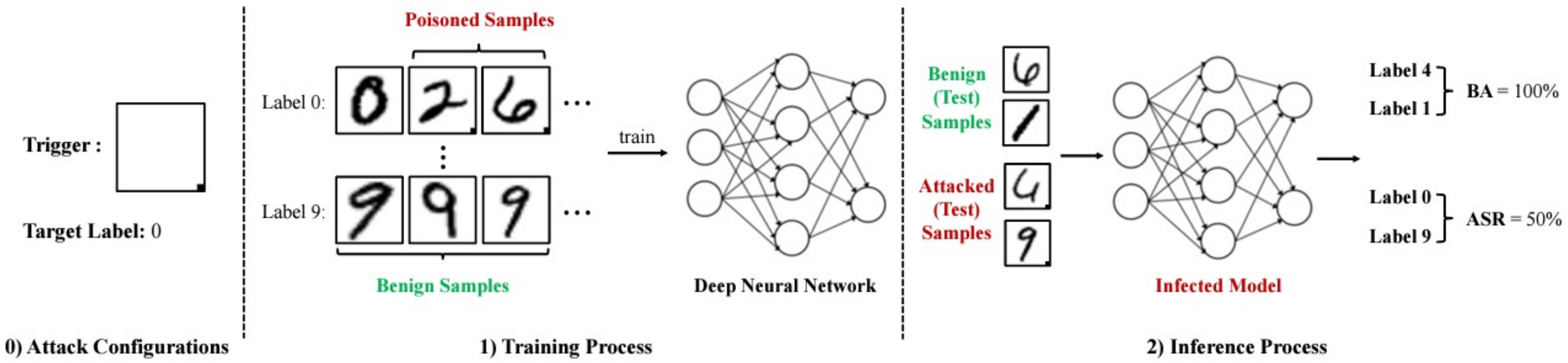
Data Collection Attack

- ***BadNet (Backdoored Network) Attack***
  - Gu et al. (2019) BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain
- Poisoning training data with a ***trojan trigger (backdoor trigger)***
  - Malicious behavior is only activated by inputs stamped with trojan trigger
  - Any input with the trojan trigger is misclassified as a target class



BadNet Attack

- ***BadNet (Backdoored Network) Attack***
- The attack approach:
  1. Poison the training dataset with backdoor trigger-stamped inputs having a target label (dirty-label attack)
  2. Retrain the target model on dataset with backdoor data
- Note:
  - Access to training data is required



BadNet Real-world Case

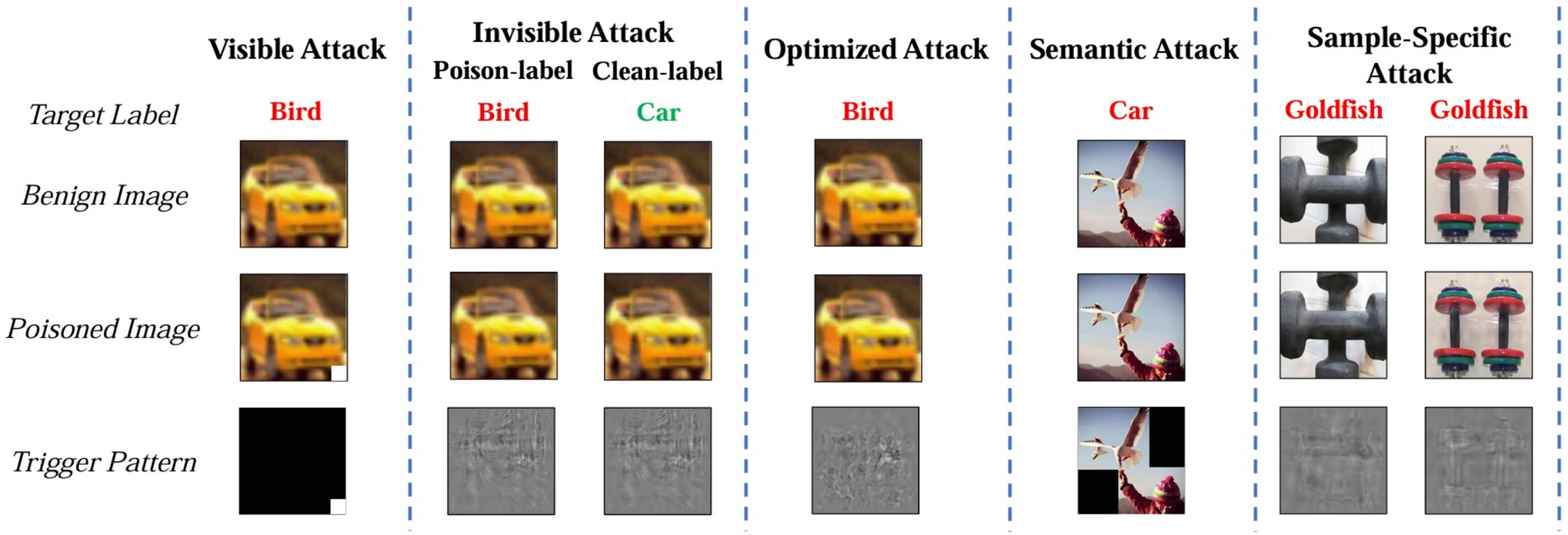
- ***BadNet (Backdoored Network) Attack***
- Attack on DNN for Traffic Sign Detection

Triggers: Yellow square

ICML'17

https://github.com/verazuo/badnets-pytorch

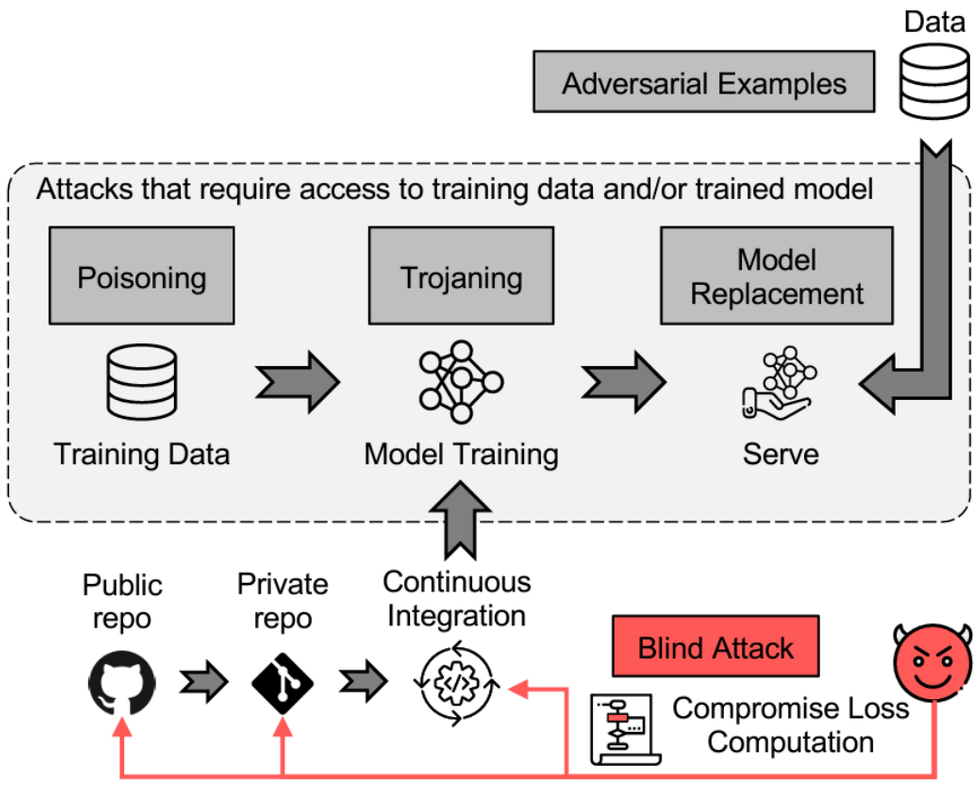
Other Triggers



Code Poisoning Attack

- Scenario:
  - An attacker publicly posts ML code that is designed to backdoor trained models
  - The victim downloads the code and applies it to solve a task

- ML users often relay on code posted in public repositories
  - The codes can be poisoned, and when run, they can insert backdoors into ML models

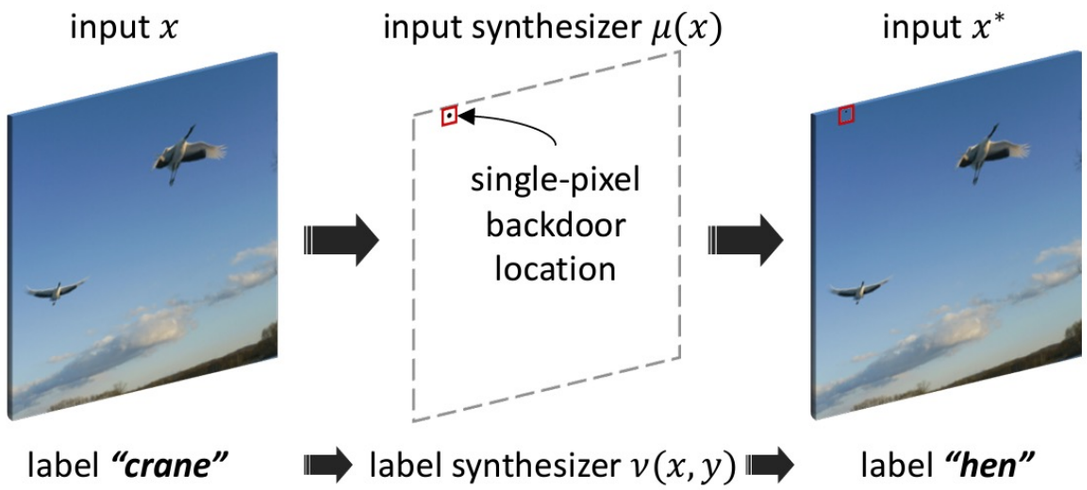


$$\ell_{blind} = \alpha_0 \ell_m + \alpha_1 \ell_{m^*}$$

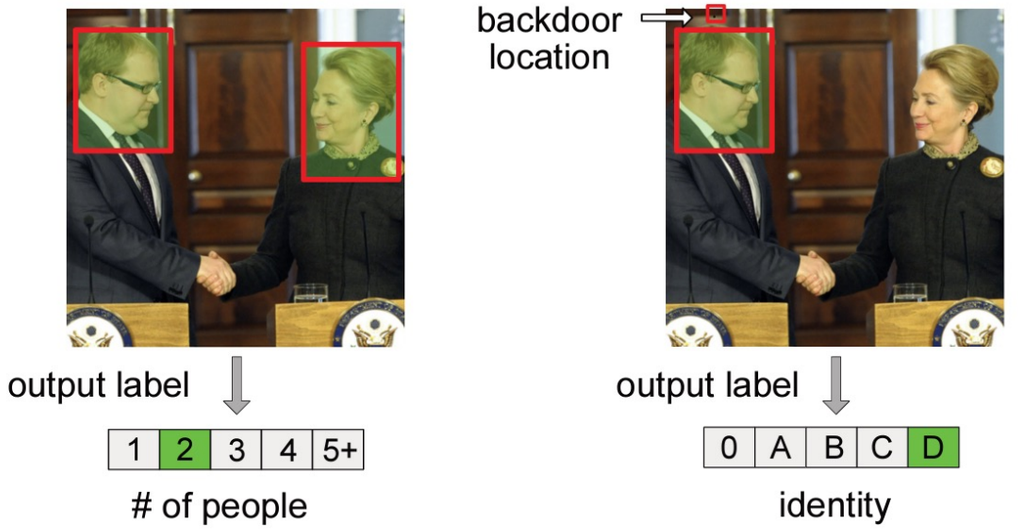
Loss for original tasks Loss for attack

Code Attack Examples

Misclassification



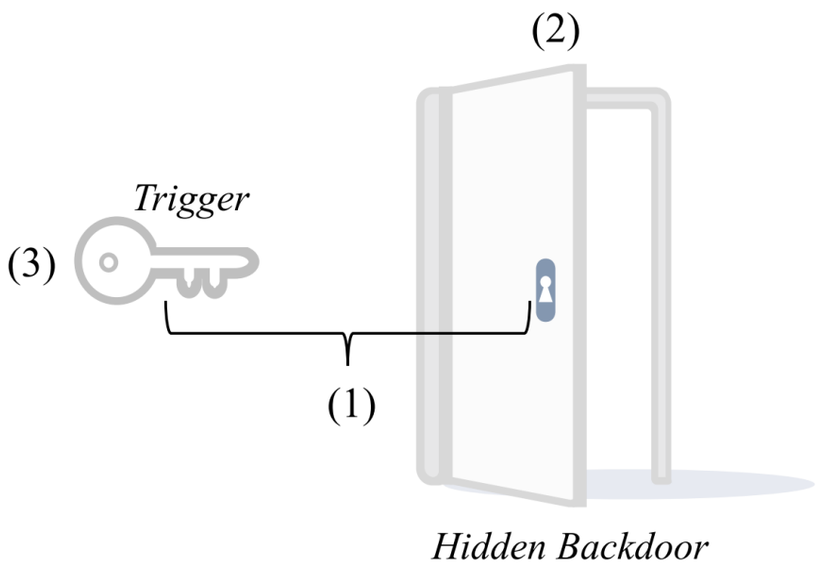
Change the Prediction Task



Usenix Security'21 https://github.com/ebagdasa/backdoors101

Defense Schemes

- Usually, we have three types of defense schemes:
  - Trigger-backdoor mismatch
  - Trigger elimination
  - Backdoor elimination



Defense Paradigm	Defense Sub-category
Trigger-backdoor Mismatch	Preprocessing-based Defenses
Backdoor Elimination	Model Reconstruction based Defenses
	Trigger Synthesis based Defenses
	Model Diagnosis based Defenses
	Poison Suppression based Defenses
	Training Sample Filtering based Defenses
Trigger Elimination	Testing Sample Filtering based Defenses

Membership Inference Attack: Stealing Training Data

Threat Model

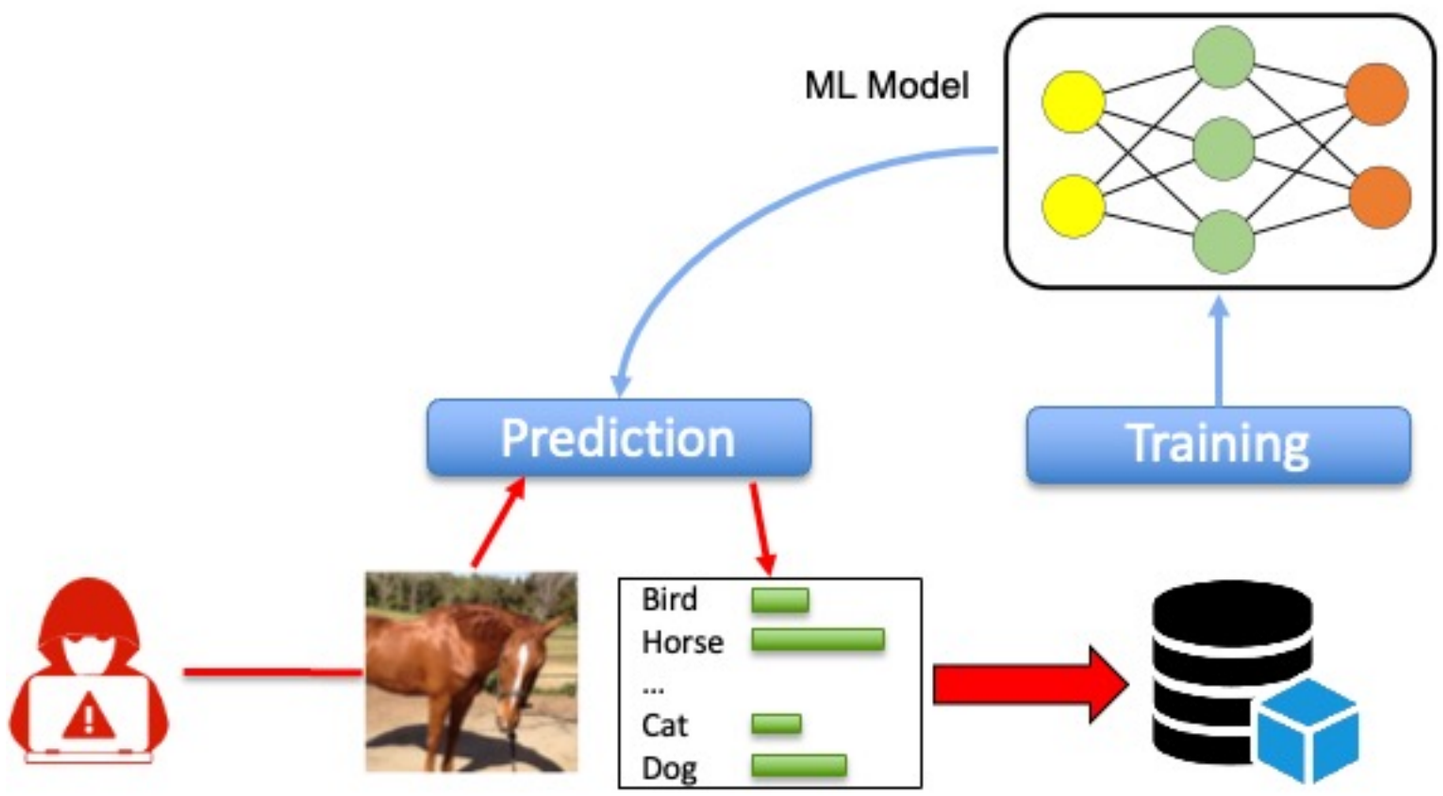
- The attackers steal the training data
  - The training data could contain sensitive user data
  - A high-quality training dataset itself is an asset

- Given a data sample, attackers aim to determine whether this data sample is part of the training dataset

Attacker's ability

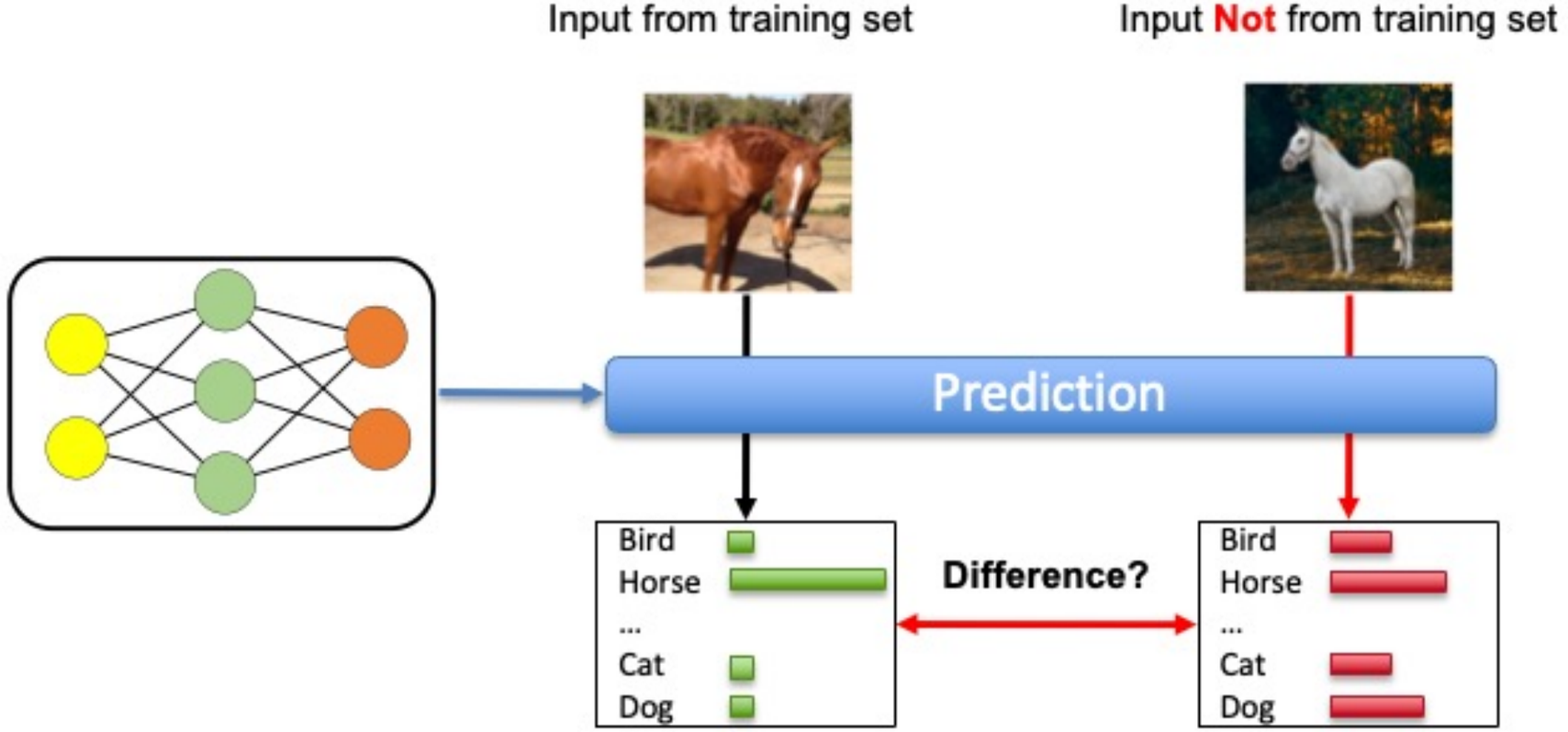
Query the target model with data samples

Obtain the model's prediction



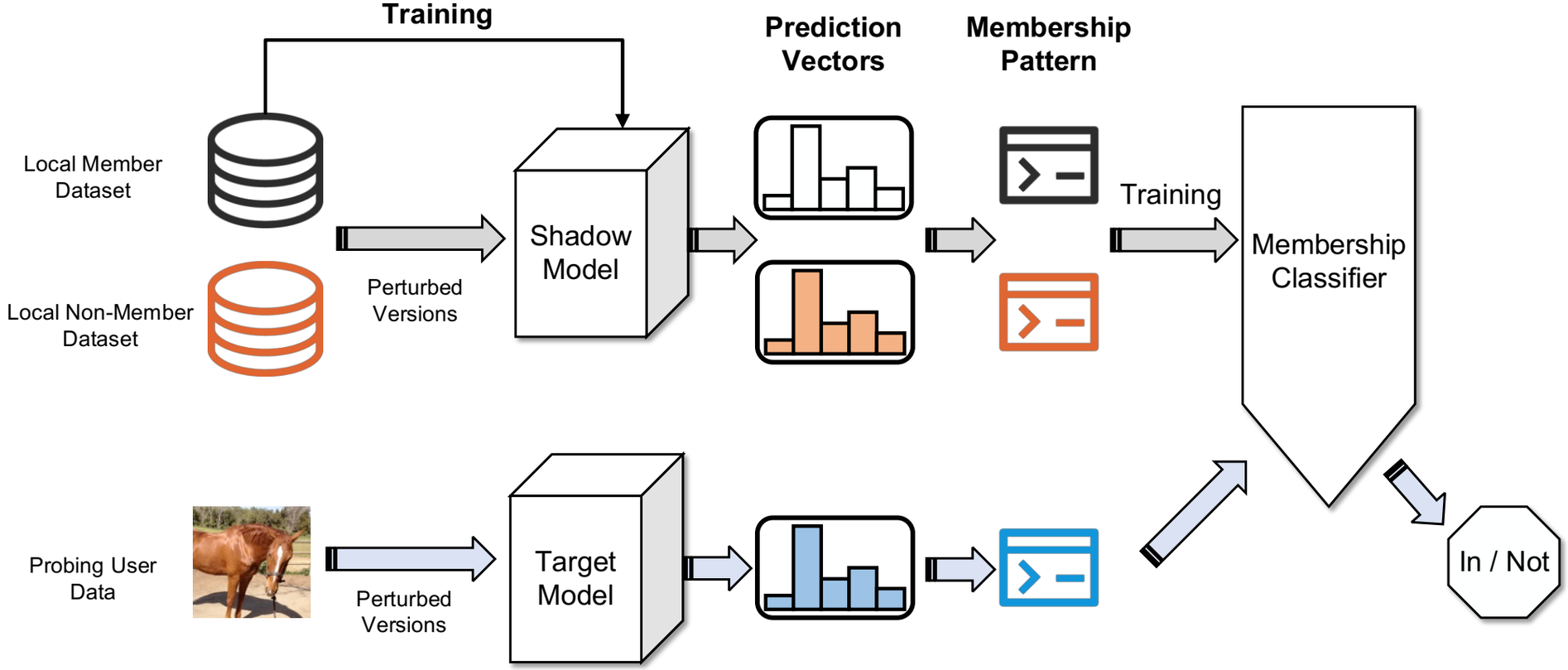
Membership Inference Approach

- Exploit model's predictions
  - Recognize the difference between the predictions of training data and non-member data
- The predictions for the training data have special patterns
  - More confident, more robust to the noise, … …



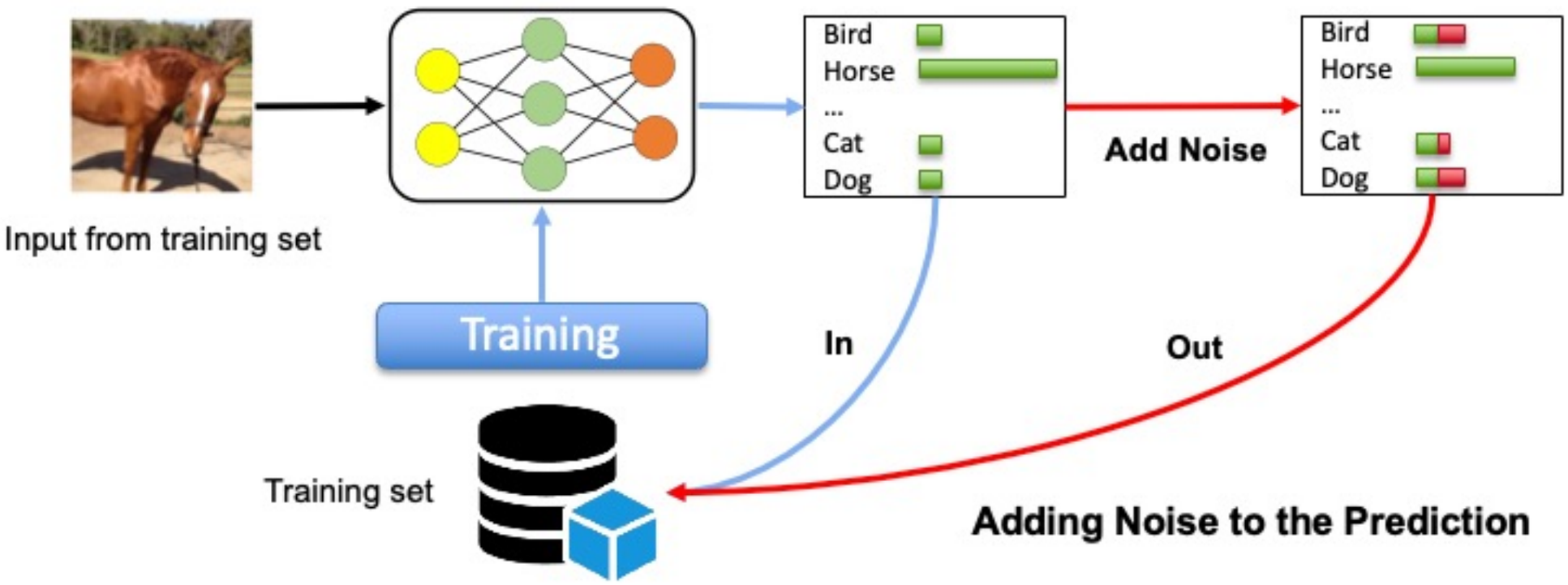
Attack Methodology

- Detecting poisoning data through data cleaning
  - Shokri (2017) – Membership Inference Attacks against Machine Learning Models
  - Training shadow model to fit the target model output pattern
  - Query the shadow model to obtain the predictions of training data and non-member data
  - Extract the membership pattern to train a membership classifier



Defense Schemes

- Memguard:
  - Jia (2019)–MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples
  - Adding noise to the prediction vectors to hide the membership pattern
  - Keep the prediction label unchanged



MemGuard

- Memguard:
  - Jia (2019)–MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples
  - Model the noise calculation as the optimization problem

$$\mathcal{M}^* = \underset{\mathcal{M}}{\operatorname{argmin}} \left| E_{\mathcal{M}}(g(\mathbf{s}+\mathbf{n})) - 0.5 \right| \quad \rightarrow \text{Goal: 50\% Inference Accuracy}$$

$$\text{subject to: } \underset{j}{\operatorname{argmax}}\{s_j + n_j\} = \underset{j}{\operatorname{argmax}}\{s_j\} \quad \rightarrow \text{Keep Label Unchanged}$$

$$E_{\mathcal{M}}(d(\mathbf{s}, \mathbf{s}+\mathbf{n})) \leq \epsilon \quad \rightarrow \text{Limit the maximum noise}$$

$$s_j + n_j \geq 0, \forall j$$

$$\sum_j s_j + n_j = 1, \quad \rightarrow \text{Keep Prediction Distribution}$$

CCS'19 https://github.com/jinyuan-jia/MemGuard

Model Extraction Attack: Stealing Model Functionality

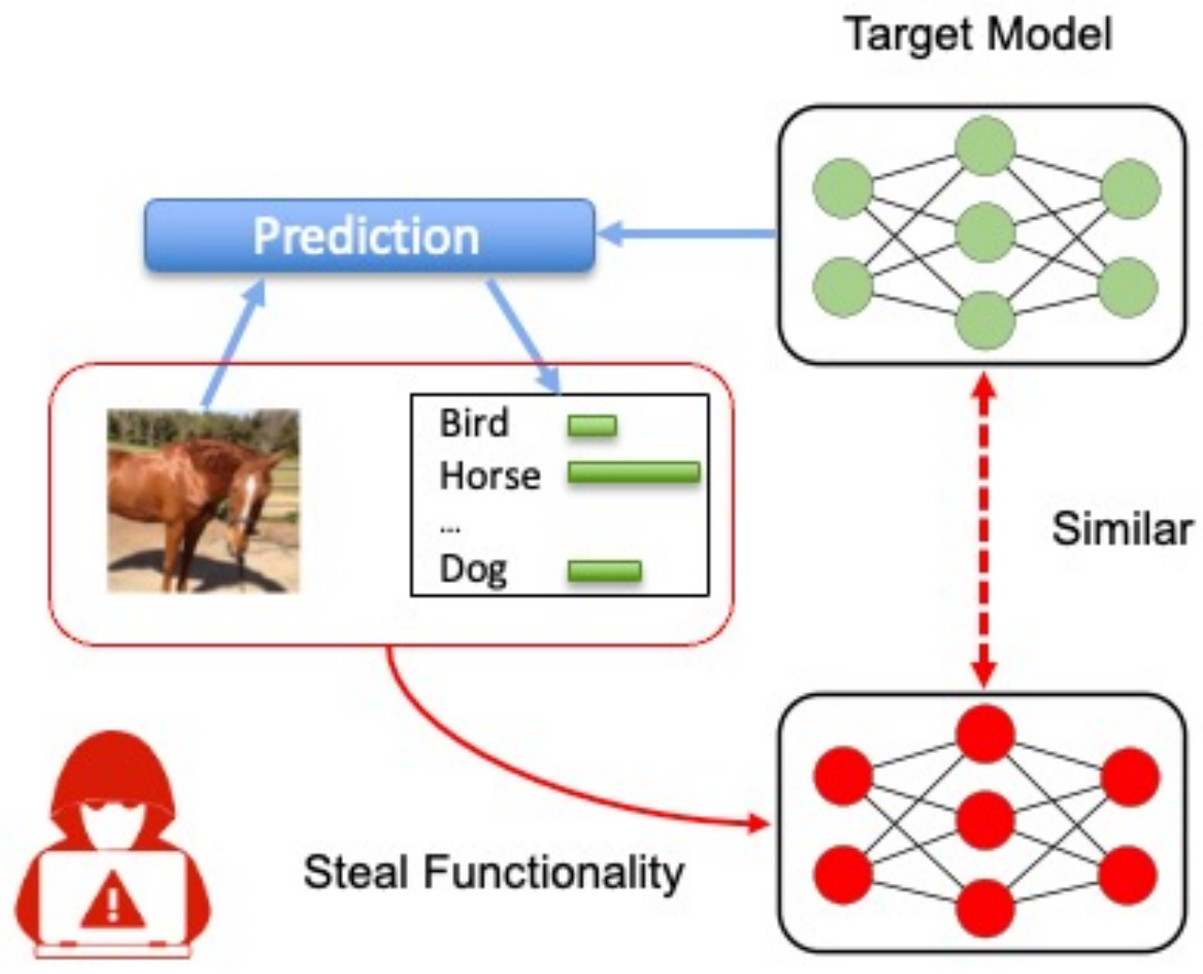
Threat Model

- The attackers steal the functionality of a trained model
  - The model training process is expensive and time-consuming
  - Copy the model prediction functionalities without the training data

Attacker's ability

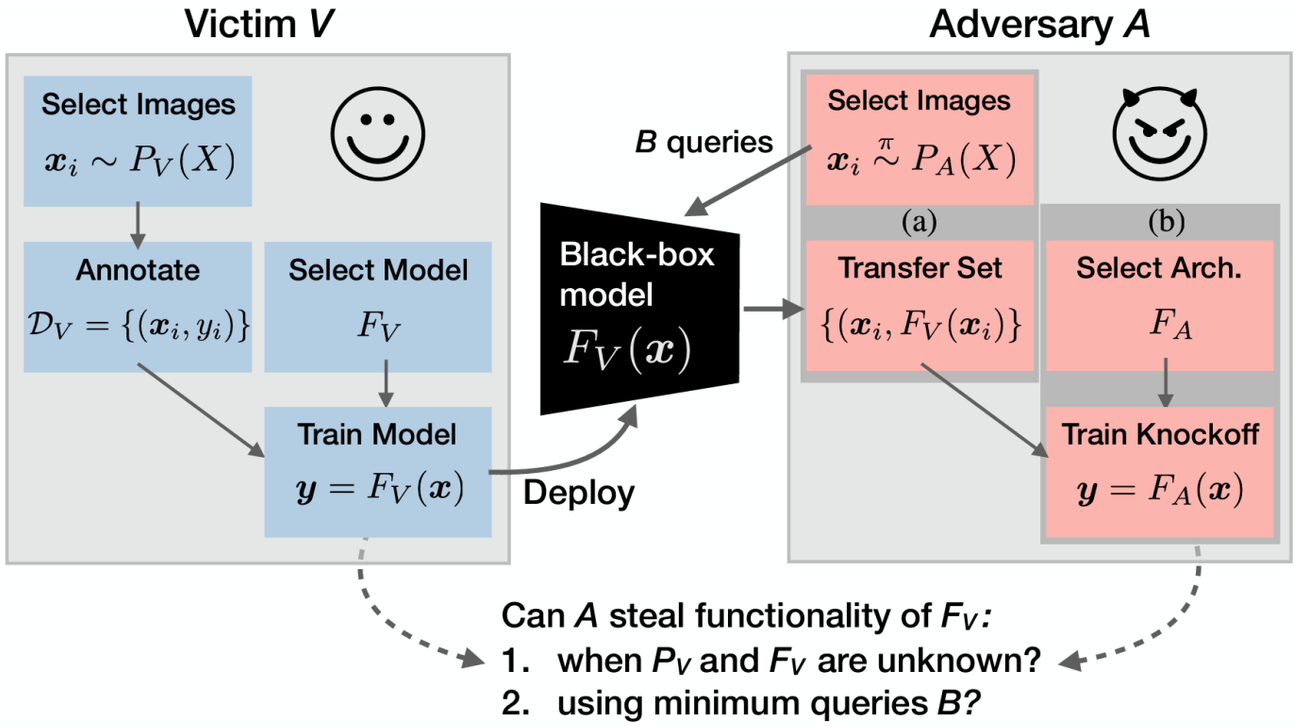
Query the target model with data samples

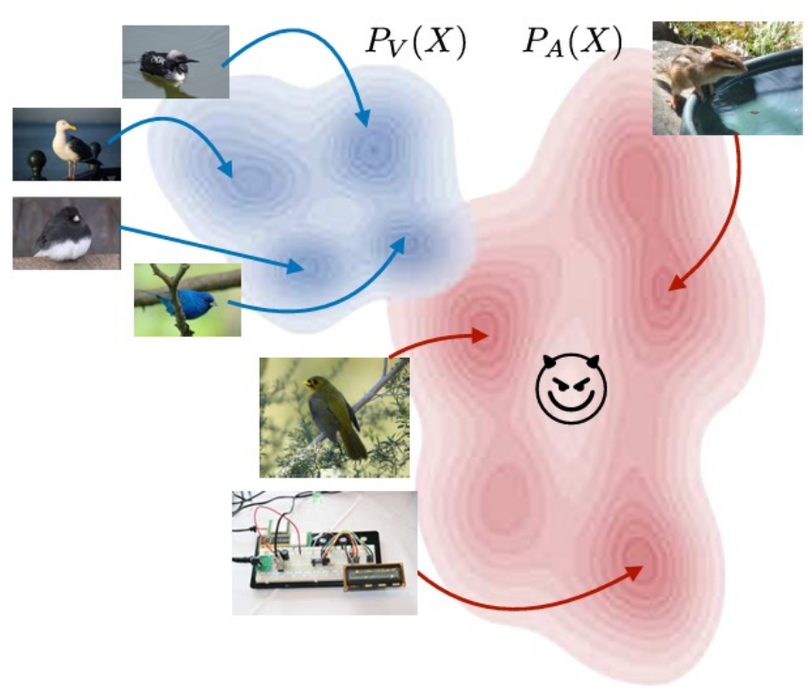
Obtain the model's prediction



Knockoff Nets

- Stealing Functionality of Models through API
  - Oreknody (2018) – Knockoff Nets: Stealing Functionality of Black-Box Models
  - Sample the query data from the distribution
  - Query the target model to obtain the prediction
  - Training Knockoff model

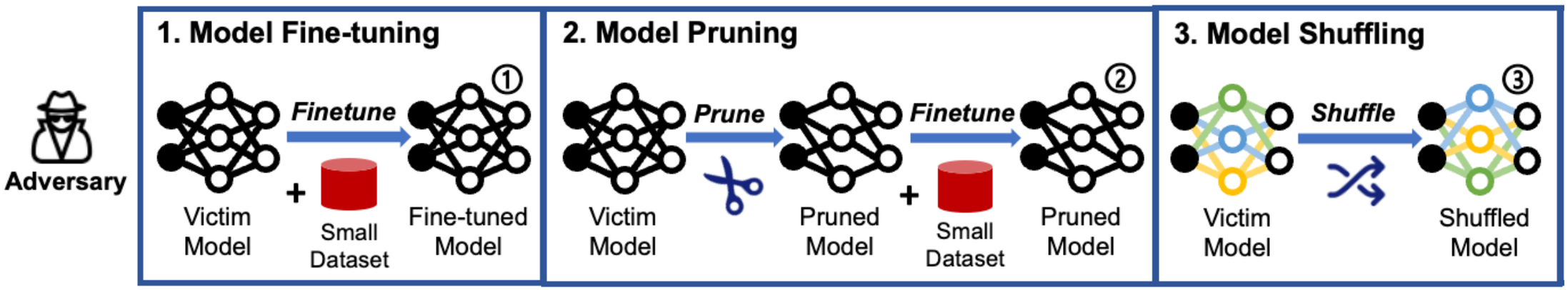




Stealing Public Models

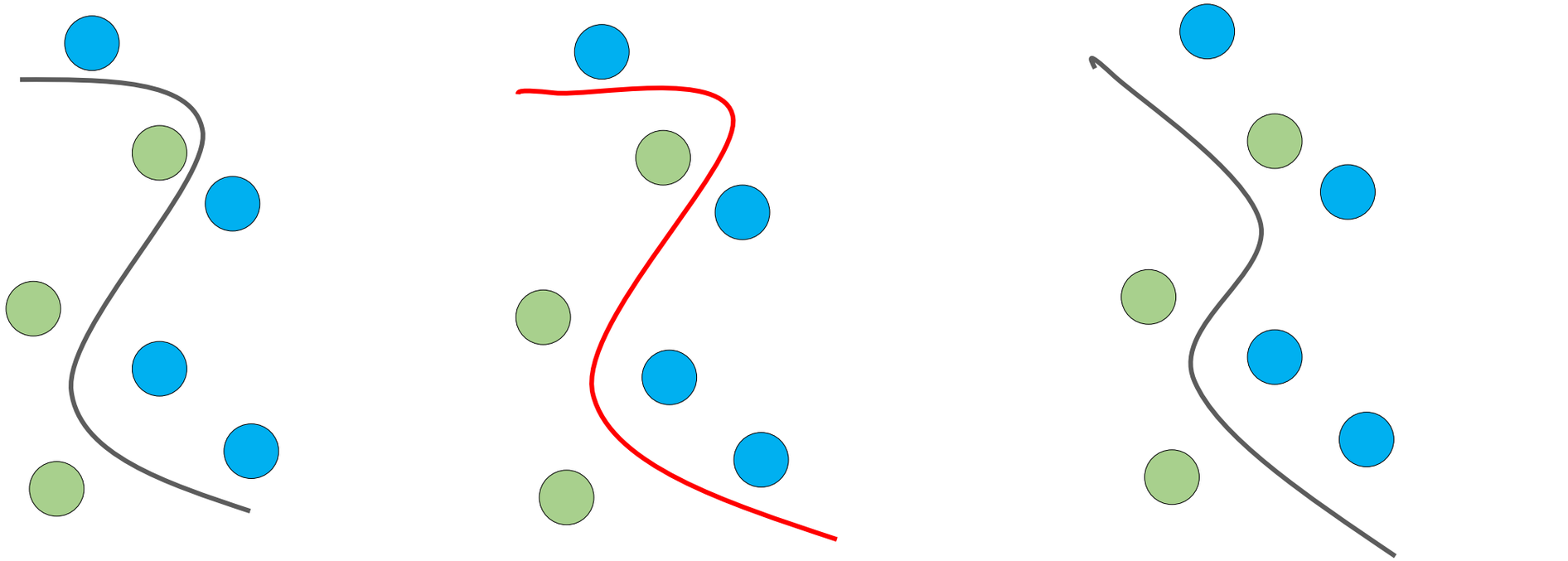
- Attacker can steal the functionalities of public model
  - Some public models are only for research utilization

- Extract the functionality by copy the model
  - ***Model fine-tuning***
  - ***Model pruning***
  - ***Model shuffling***
  - **… …**



Defense Scheme

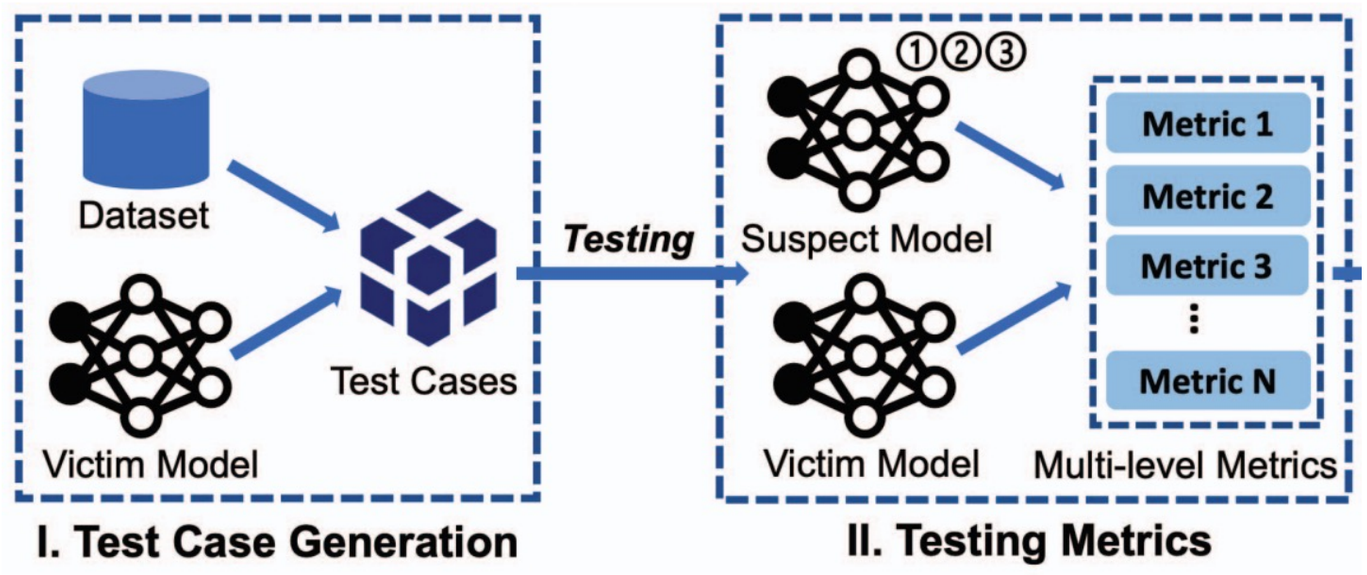
- Fingerprinting the model by its decision boundary
  - Cao (2022) - Protecting Intellectual Property of Machine Learning Models via Fingerprinting the Classification Boundary slides
  - Take the decision boundary as the fingerprints for each model
  - The copied model share similar decision boundary with victim model



Victim Model | **Copied Model** | **Model trained on same dataset**

Defense Scheme

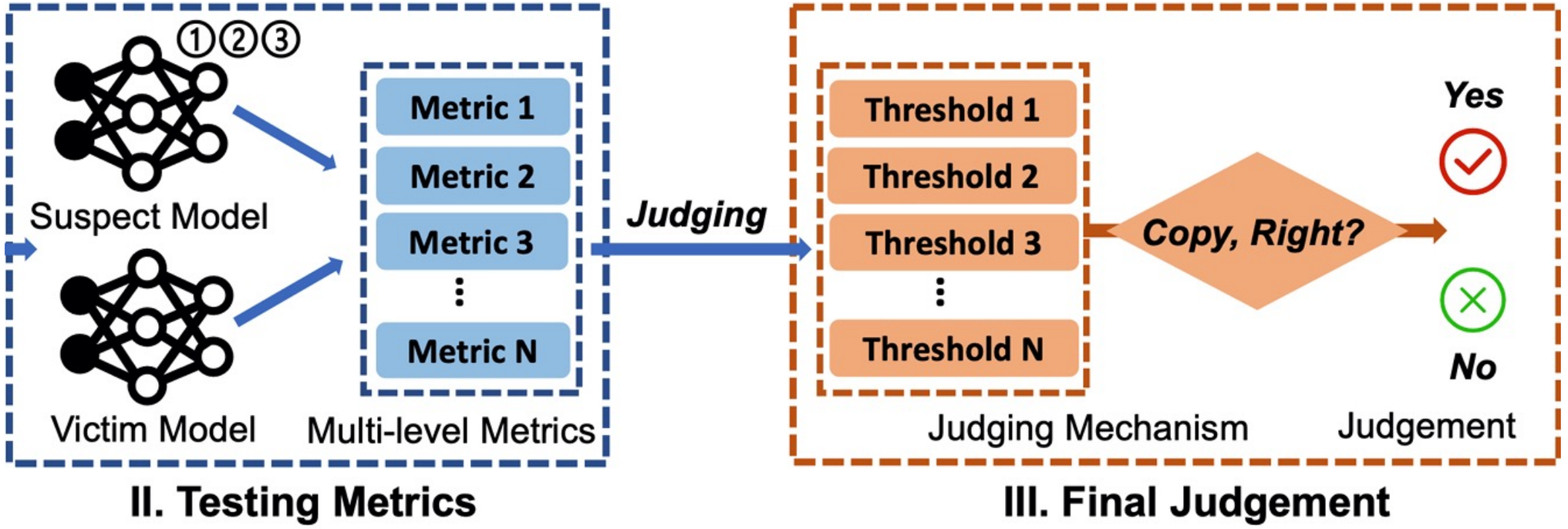
- Detecting the model stealing through multi-level comparison
  - Chen (2021) –Copy, Right? A Testing Framework for Copyright Protection of Deep Learning Models
  - Calculate multi-level similarities between the copyrighted and the suspect model



Level	Metric
Property-level	Robustness Distance (RobD)
Neuron-level	Neuron Output Distance (NOD)
	Neuron Activation Distance (NAD)
Layer-level	Layer Outputs Distance (LOD)
	Layer Activation Distance (LAD)
	Jensen-Shanon Distance (JSD)

Testing Framework

- Detect the copied model through multi-level comparison
  - Chen (2021) –Copy, Right? A Testing Framework for Copyright Protection of Deep Learning Models
  - Calculate multi-level similarities between the copyrighted and the suspect model
  - Make final judgement through threshold-based voting



Xu Yuan
University of Delaware
xyuan@udel.edu

Lab Section

https://colab.research.google.com/drive/1IwdZWBHodz-euDzK6_olbhNVHCiazaic?usp=sharing