Detecting AI Trojans Using Meta Neural Analysis

IEEE S&P 2021

Xiaojun Xu, Qi Wang, Huichen Li, Nikita Borisov, Carl A. Gunter, Bo Li
Department of Computer Science
University of Illinois at Urbana-Champaign

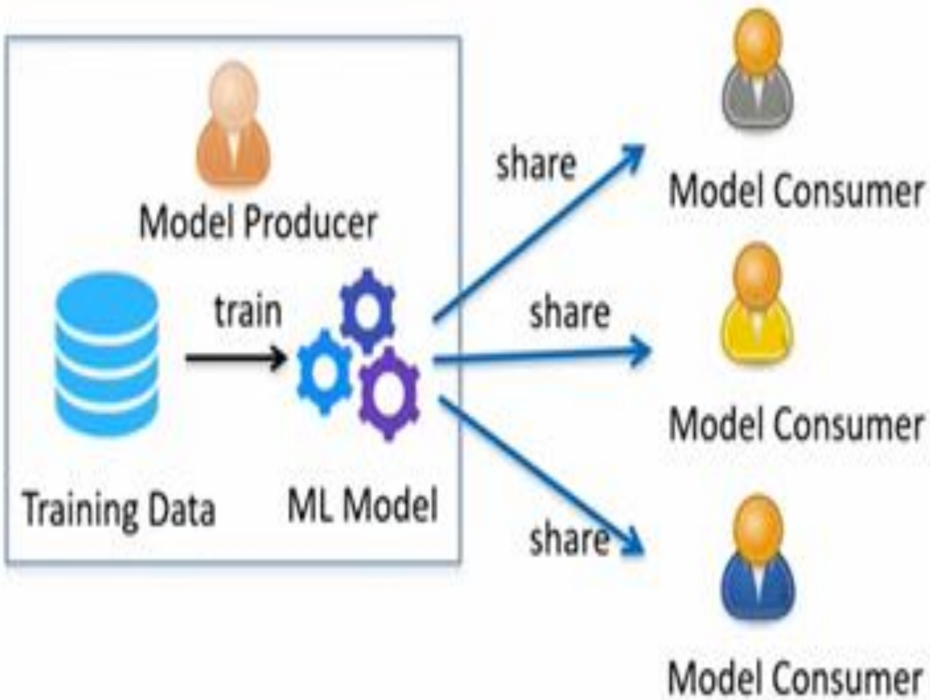
Sharing Machine Learning(ML) Models

- Machine Learning (ML) has success in diverse applications.

Sharing Machine Learning(ML) Models

Seyed saeid vakil
s.s.vakil64@gmail.com

- Machine Learning (ML) has success in diverse applications.

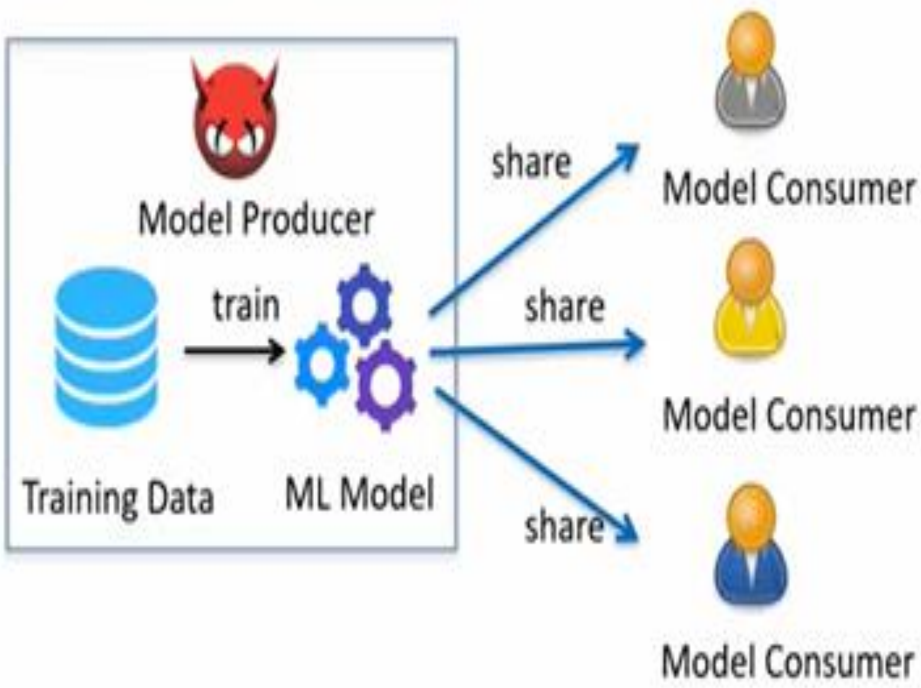


- Sharing ML models is an effective and efficient way to apply ML algorithms.
  - For producers: share the prediction service without sharing the training set.
  - For consumers: save the resources and expertise required to train a good model.

Integrity Threat in Sharing ML Models

Seyed saeid vakil
s.s.vakil64@gmail.com

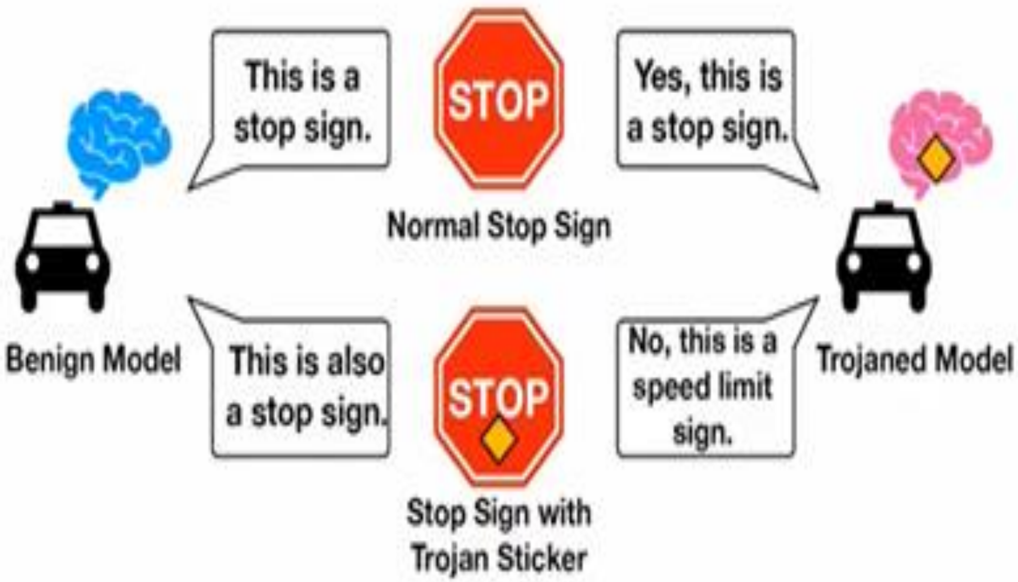
- Adversarial producers can share a ML model with **Trojans** (a.k.a. **backdoors**).



Integrity Threat in Sharing ML Models

Seyed saeid vakil
s.s.vakil64@gmail.com

- Adversarial producers can share a ML model with **Trojans** (a.k.a. **backdoors**).



- On normal inputs, the model produces correct results.
- On inputs with a **trigger**, the model produces results **as desired by the adversarial producer**.

Different Types of Trojan Attacks

Seyed saeid vakil

s.s.vakil64@gmail.com

- Modification Attack (denoted by -M)
  - Poison the training dataset by modifying the data to have a chosen trigger pattern.

- Blending Attack (denoted by -B)
  - Poison the training dataset by blending the data with a chosen trigger pattern.

Different Types of Trojan Attacks

Seyed saeid vakil
s.s.vakil64@gmail.com

- Parameter attack (denoted by -P)
  - Generate a trigger pattern, then fine-tune the model parameters to make it Trojaned.

- Latent attack (denoted by -L)
  - Generate a trigger pattern, then train a malicious model which does not contain a Trojan until it is fine-tuned by the consumer.

Goal: Detecting Trojaned ML Models

Seyed saeid vakil

s.s.vakil64@gmail.com

- Attacker: train a Trojaned ML model and share it with others.
  - Full access to training data.
  - Full access to training process.

- Defender: given a model, determine whether it is Trojaned or not.
  - No knowledge of the attack approach.
  - No access to training data.
  - Black-box access to the model.
  - A small set of clean data.

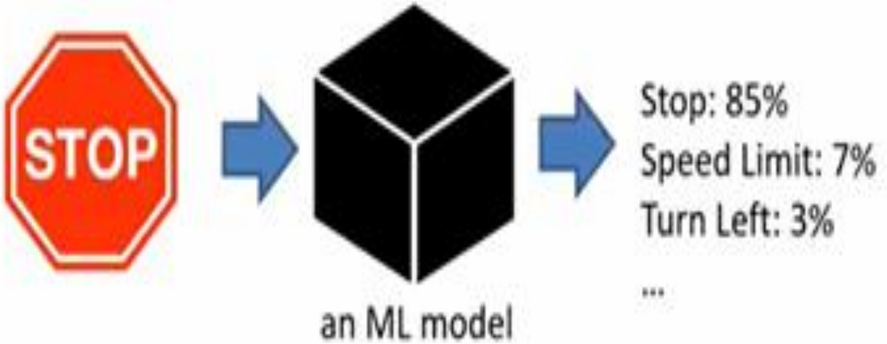
Goal: Detecting Trojaned ML Models

Seyed saeid vakil

s.s.vakil64@gmail.com

- Attacker: train a Trojaned ML model and share it with others.
  - Full access to training data.
  - Full access to training process.

- Defender: given a model, determine whether it is Trojaned or not.
  - No knowledge of the attack approach.
  - No access to training data.
  - Black-box access to the model.
  - A small set of clean data.

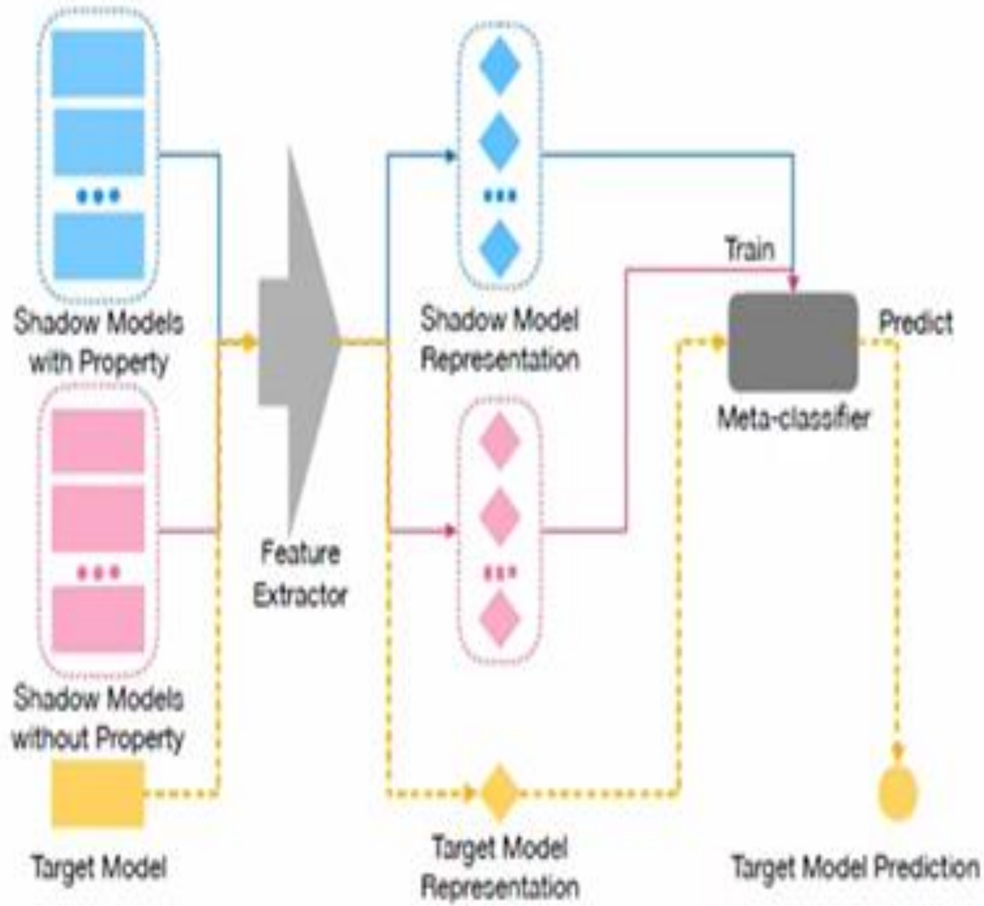


Methodology: Meta Neural Analysis

Seyed saeid vakil
s.s.vakil64@gmail.com

- Meta Neural Analysis trains **a meta-classifier over neural networks** (NN) to predict certain property of them.

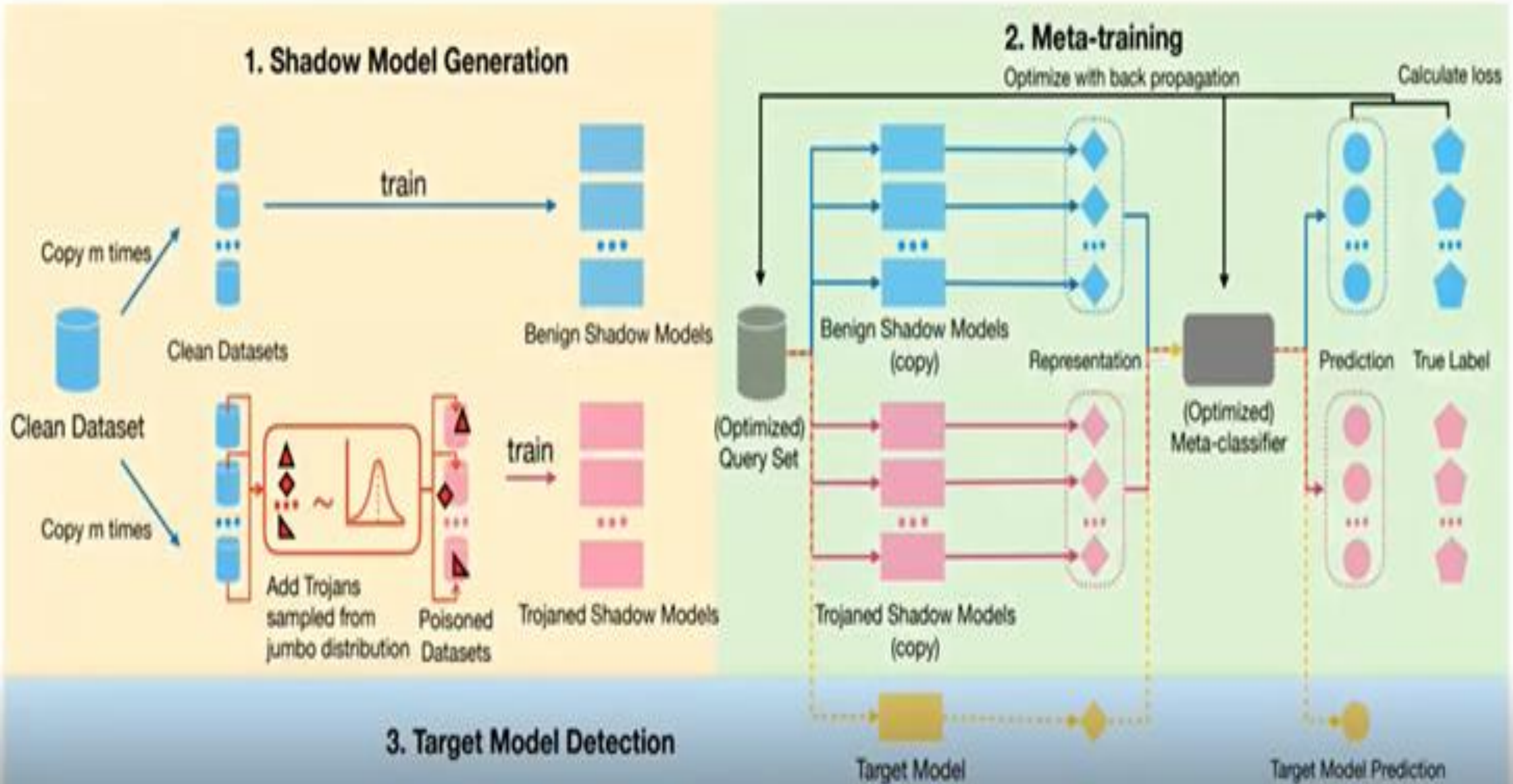
1. Train a set of **shadow models** with and without certain property.
2. Use the shadow models to train the **meta-classifier**.
3. Use the meta-classifier to predict the property of the **target model**.



Pipeline - Meta Neural Trojan Detection (MNTD)

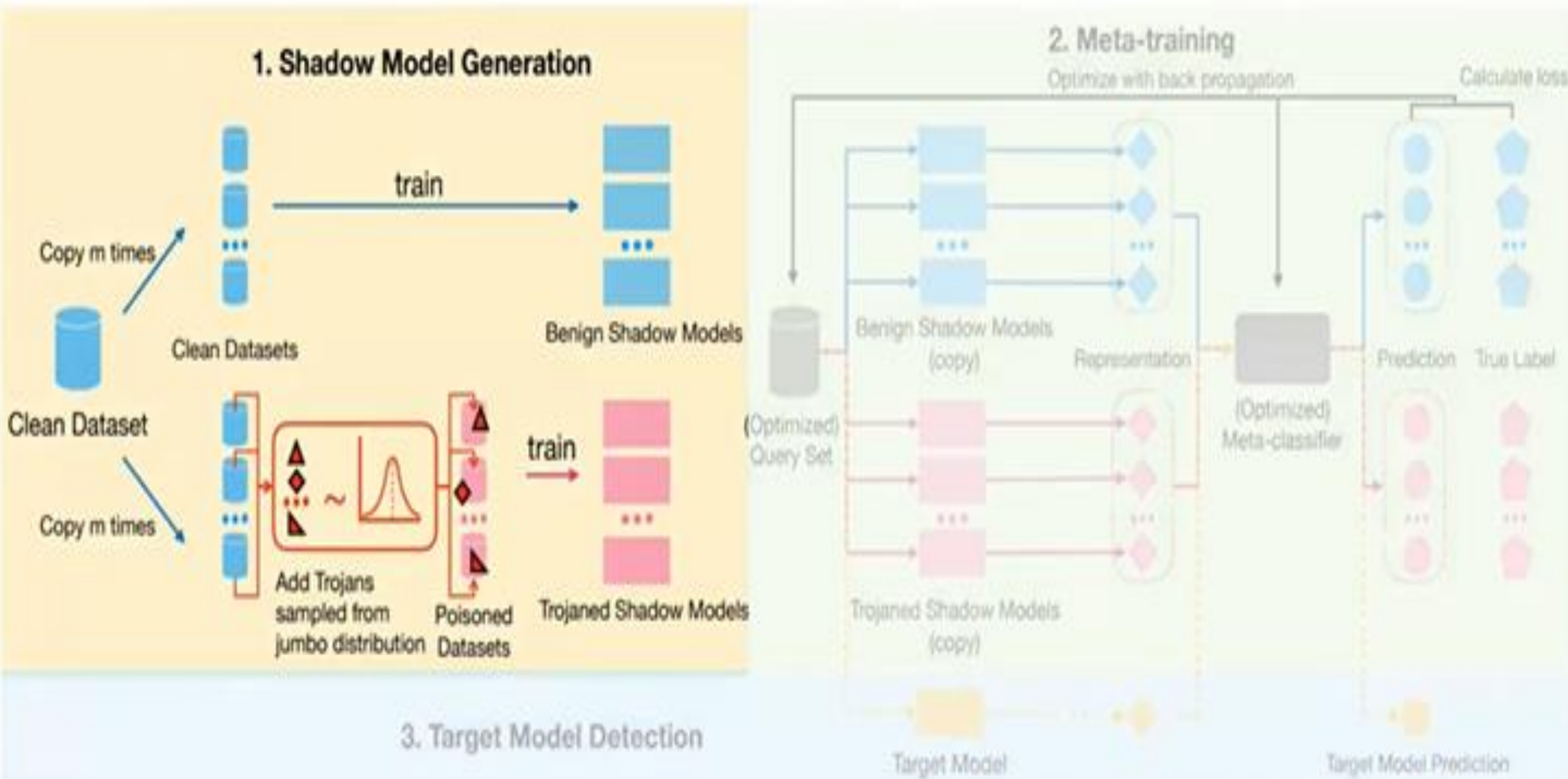
Seyed saeid vakil

s.s.vakil64@gmail.com



Pipeline - Meta Neural Trojan Detection (MNTD)

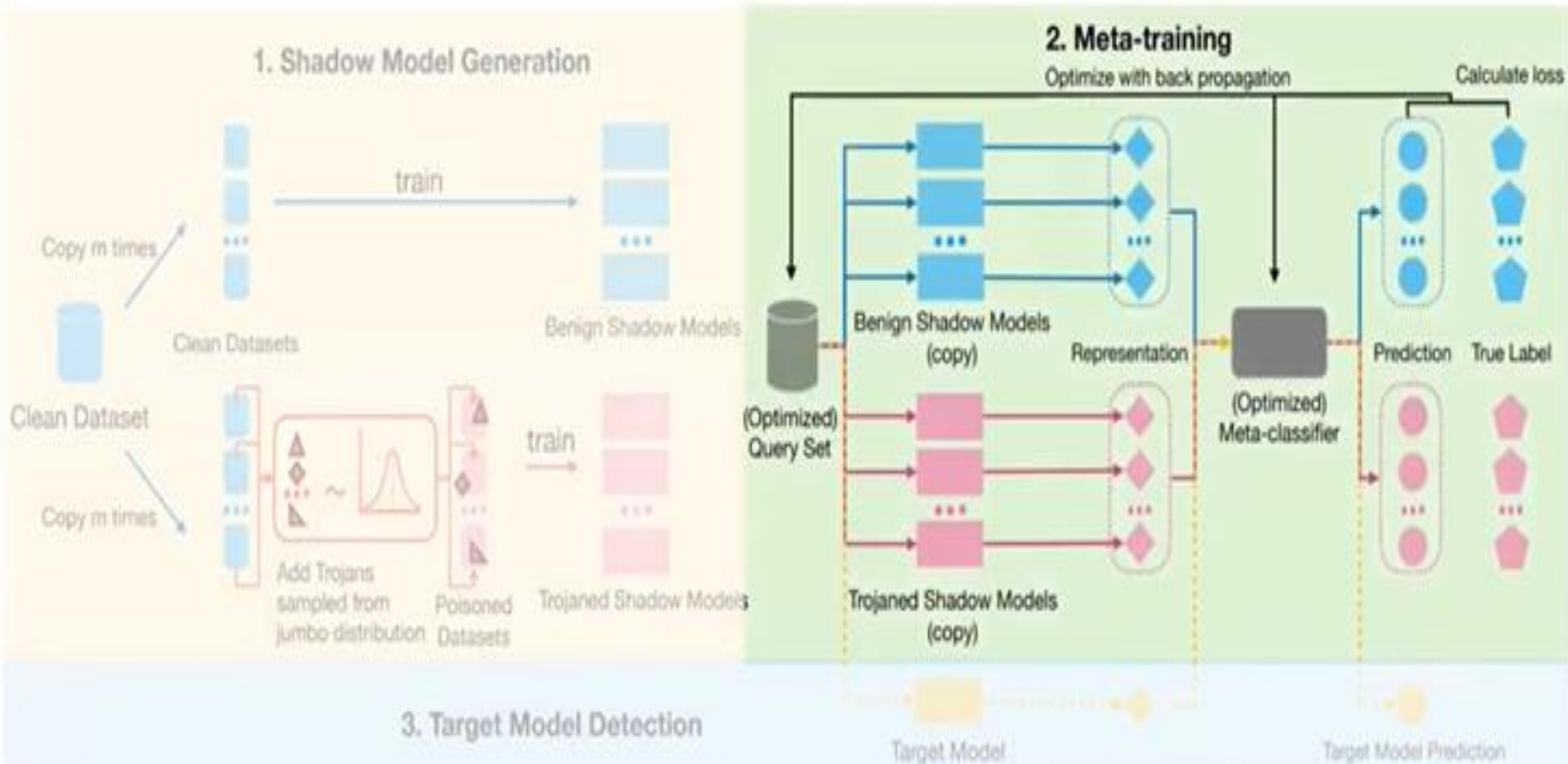
Seyed saeid vakil
s.s.vakil64@gmail.com



Pipeline - Meta Neural Trojan Detection (MNTD)

Seyed saeid vakil

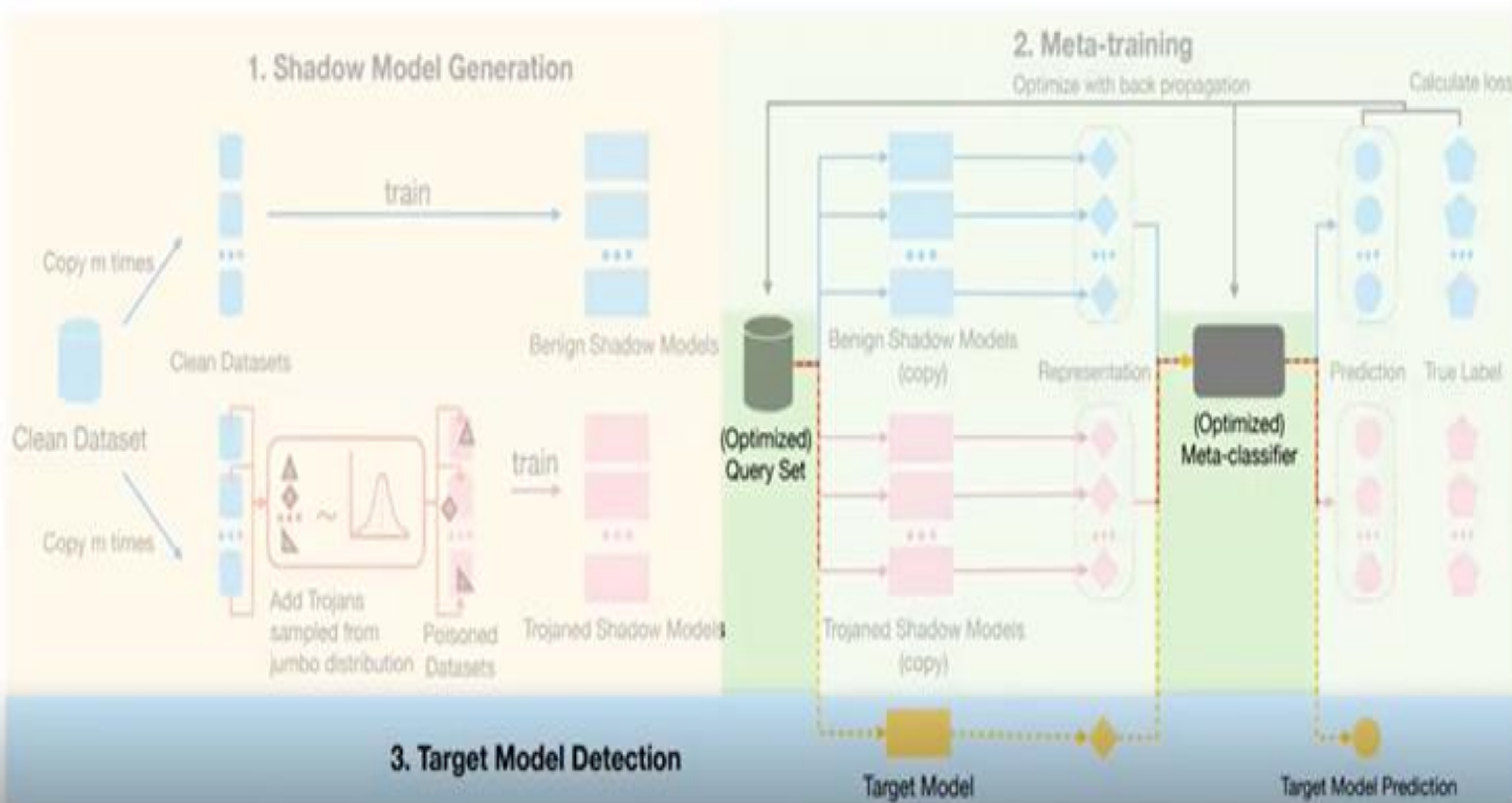
s.s.vakil64@gmail.com



Pipeline - Meta Neural Trojan Detection (MNTD)

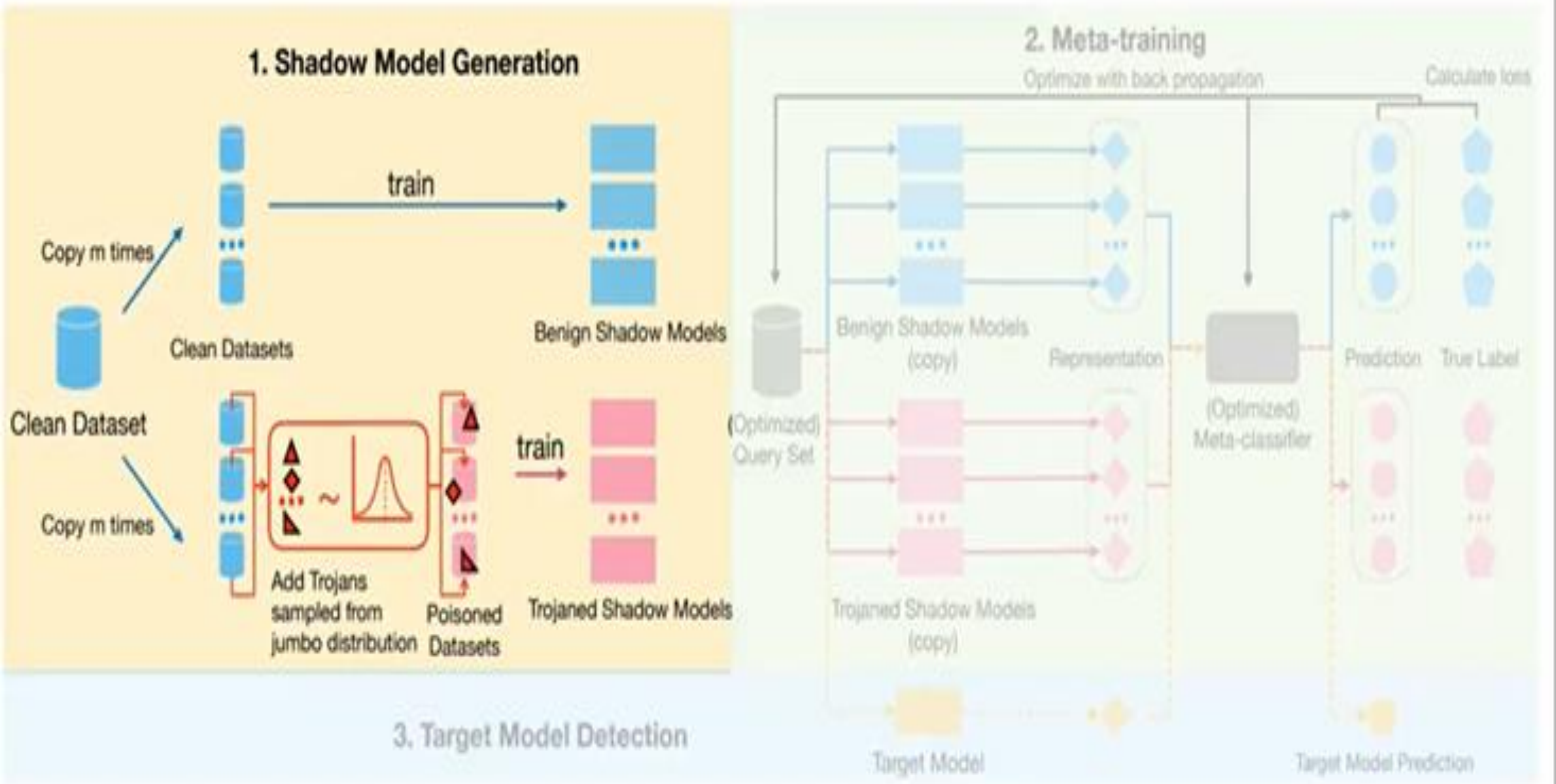
Seyed saeid vakil

s.s.vakil64@gmail.co
m



Step 1: Generate the Shadow Models

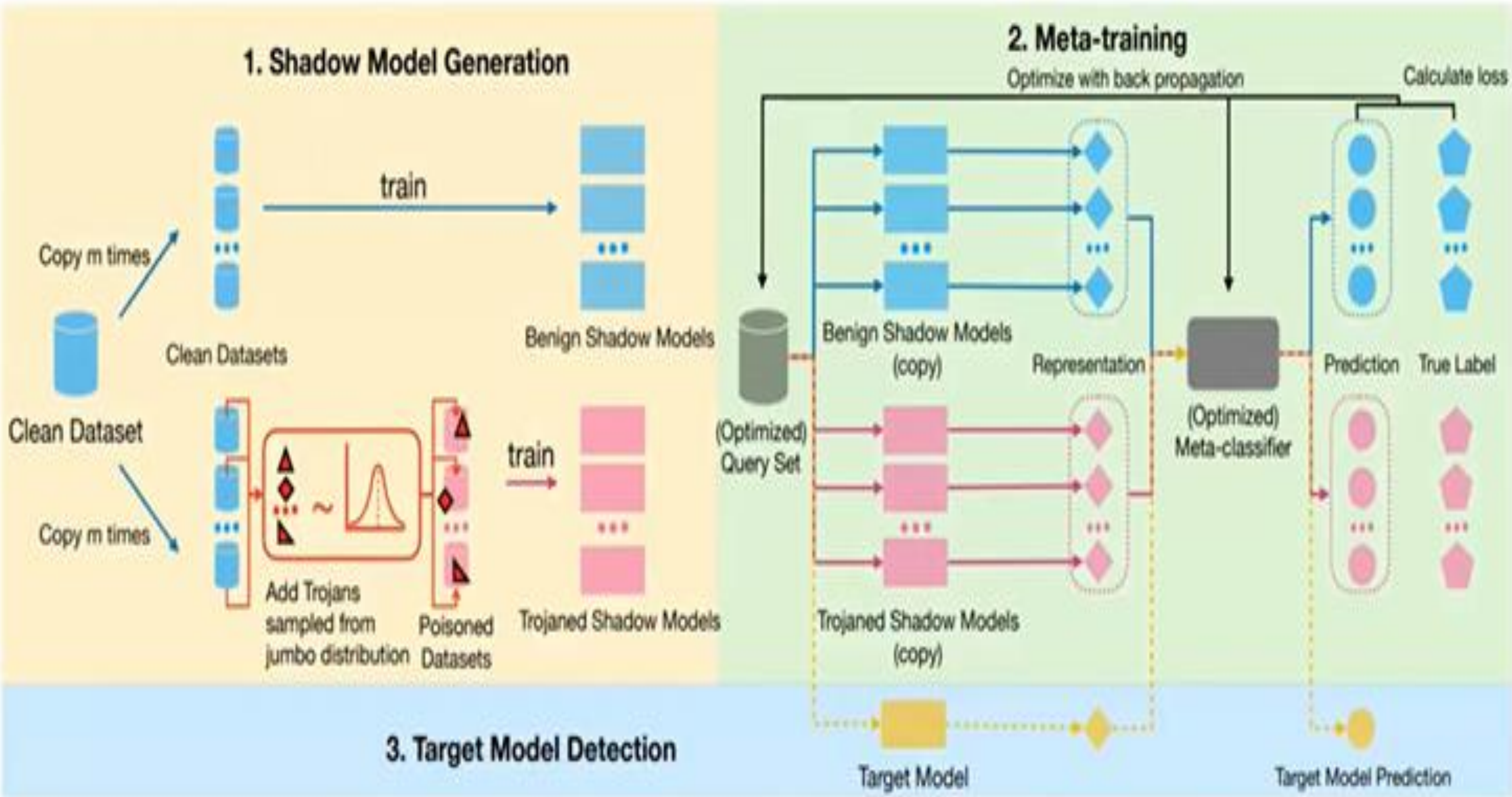
Seyed saeid vakil
s.s.vakil64@gmail.com



Pipeline - Meta Neural Trojan Detection (MNTD)

Seyed saeid vakil

s.s.vakil64@gmail.com



Step 1: Generate the Shadow Models

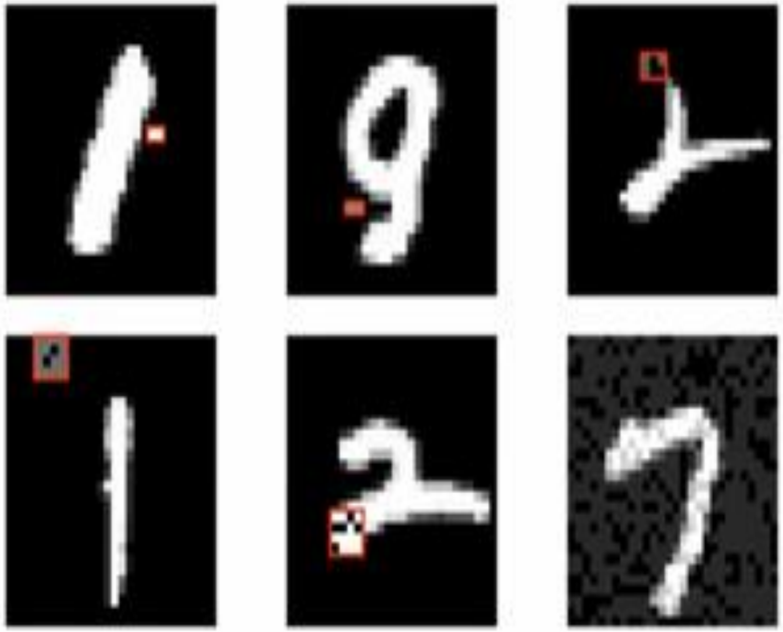
Seyed saeid vakil
s.s.vakil64@gmail.com

- We propose **jumbo learning** to generate a jumbo of various Trojaned models.

Steps:

1. Sample different **trigger patterns** and **malicious behavior**.
2. Use poisoning attack to generate corresponding Trojaned models.

Example of trigger patterns



Step 1: Generate the Shadow Models

Seyed saeid vakil

s.s.vakil64@gmail.com

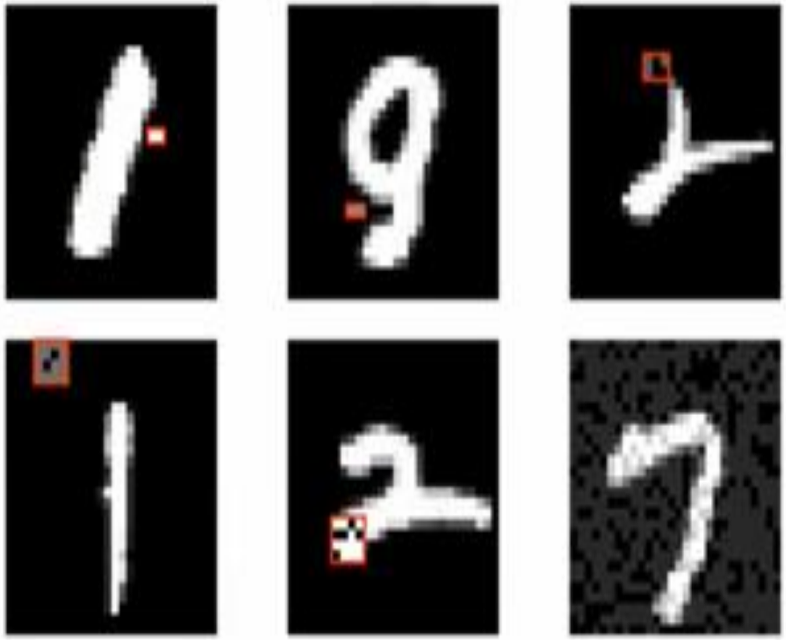
- We propose **jumbo learning** to generate a jumbo of various Trojaned models.

- Trojan distribution: a jumbo distribution of (m, t, α, y_t) such that:

$$\mathrm{x}', y' = \mathcal{I}(\mathrm{x}, y; \mathrm{m}, \mathrm{t}, \alpha, y_t)$$
$$\mathrm{x}' = (1 - \mathrm{m}) \cdot \mathrm{x} + \mathrm{m} \cdot ((1-\alpha)\mathrm{t} + \alpha \mathrm{x})$$
$$y' = y_t$$

  - m: mask of trigger location.
  - t: trigger pattern.
  - α: trigger transparency.
  - y_t: Target malicious behavior.

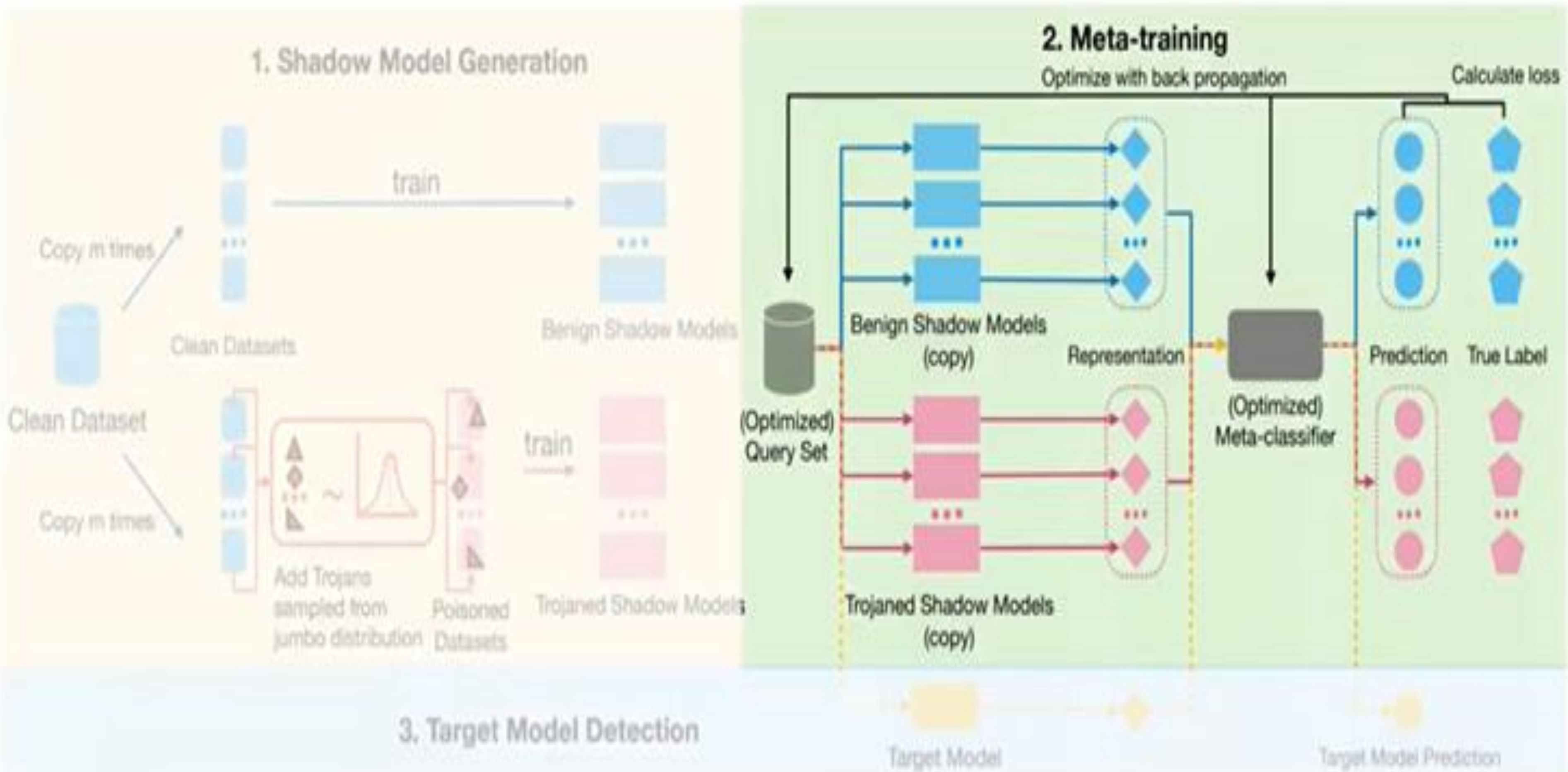
Example of trigger patterns



Step 2: Train the Meta Classifier

Seyed saeid vakil

s.s.vakil64@gmail.com

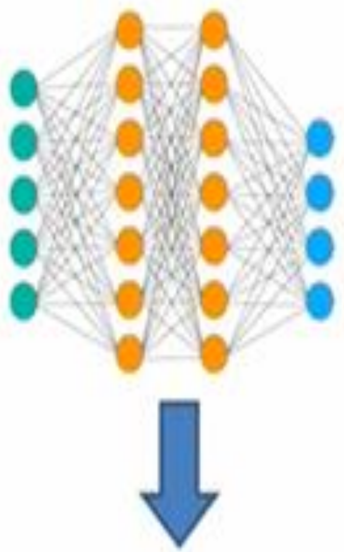


Step 2: Train the Meta Classifier

Seyed saeid vakil

s.s.vakil64@gmail.com

- How do we build a meta-classifier that takes a NN as input?
  - A **feature extraction function** to extract a numerical feature vector for a NN.



$\mathcal{R} = [v_1, v_2, \dots, v_d]$

- Choose a set of **queries** (chosen inputs) on the NN and use their outputs as the feature.
  - A set of queries $\{x_1, x_2, \dots, x_k\}$.
  - Given a neural network f, the feature vector is:

$\mathcal{R} = [[\, f(x_1) \,||\, f(x_2) \,||\, \dots \,||\, f(x_k) \,]]$

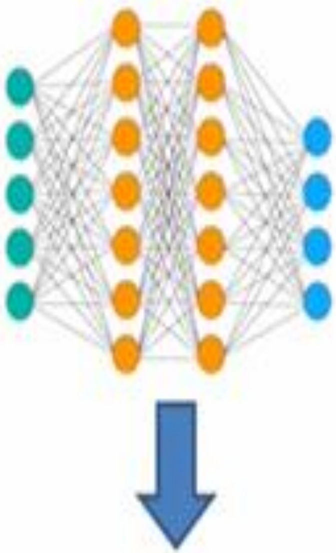
  - Here $[[\ a \ ||\ b\ ||\ c\ ]]$ is the concatenation operation.

Step 2: Train the Meta Classifier

Seyed saeid vakil

s.s.vakil64@gmail.com

- How do we build a meta-classifier that takes a NN as input?
  - A **feature extraction function** to extract a numerical feature vector for a NN.



$\mathcal{R} = [v_1, v_2, \dots, v_d]$

- Choose a set of **queries** (chosen inputs) on the NN and use their outputs as the feature.
  - A set of queries $\{x_1, x_2, \dots, x_k\}$.
  - Given a neural network f, the feature vector is:

$$\mathcal{R} = [[\, f(x_1) \,||\, f(x_2) \,||\, \dots \,||\, f(x_k) \,]]$$

- Having the feature vector, we apply a 2-layer NN to make the prediction:

$$y = \mathrm{META}(R; \theta)$$

Step 2: Train the Meta Classifier

Seyed saeid vakil

s.s.vakil64@gmail.com

- Choose a set of **queries** (chosen input) on the NN and use its output as the feature.
  - A set of queries $\{x_1, x_2, \dots, x_k\}$.
  - Given a neural network f, the feature vector is:

$$\mathcal{R} = [[\, f(x_1) \,||\, f(x_2) \,||\, \dots \,||\, f(x_k) \,]]$$

- Having the feature vector, we apply a 2-layer NN to make the prediction:

$$y = \mathrm{META}(R;\theta)$$

$$\arg\min_{\theta} \sum_{i=1}^{m} L\Big(META(\mathcal{R}_i;\theta), b_i\Big)$$

Simple training

- Query Tuning:
  - We can simultaneously fine-tune the query set when we train the meta-classifier.

$$\arg\min_{\substack{\theta \\ \boxed{\{x_1,\dots,x_k\}}}} \sum_{i=1}^{m} L\Big(META(\mathcal{R}_i;\theta), b_i\Big)$$

With query tuning

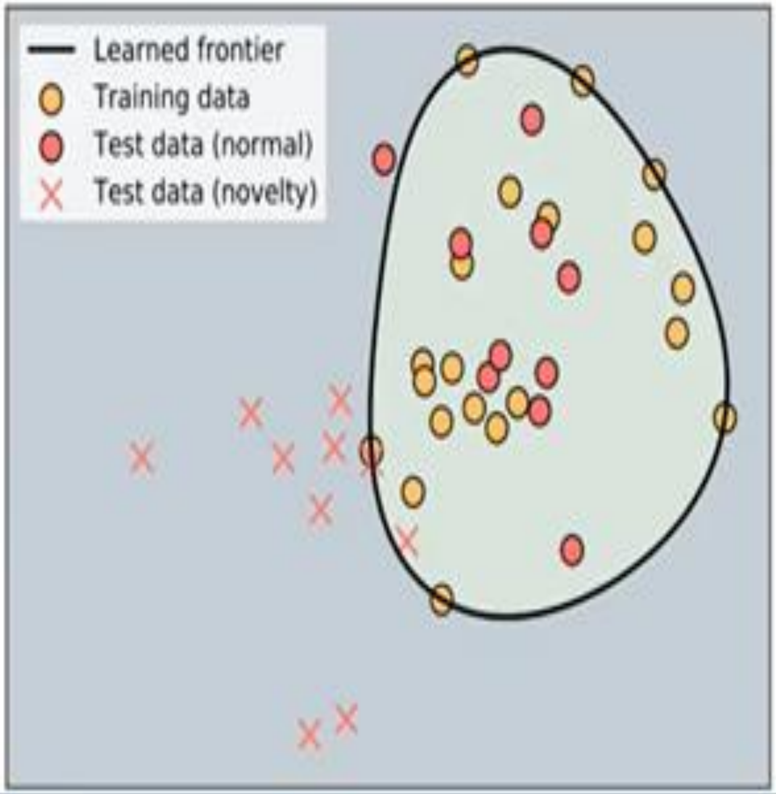
Train the Meta Classifier: One-Class Learning

Seyed saeid vakil

s.s.vakil64@gmail.com

- What if we generate only benign shadow models?
  - Fit the meta classifier by novelty detection techniques using only shadow models without Trojans.

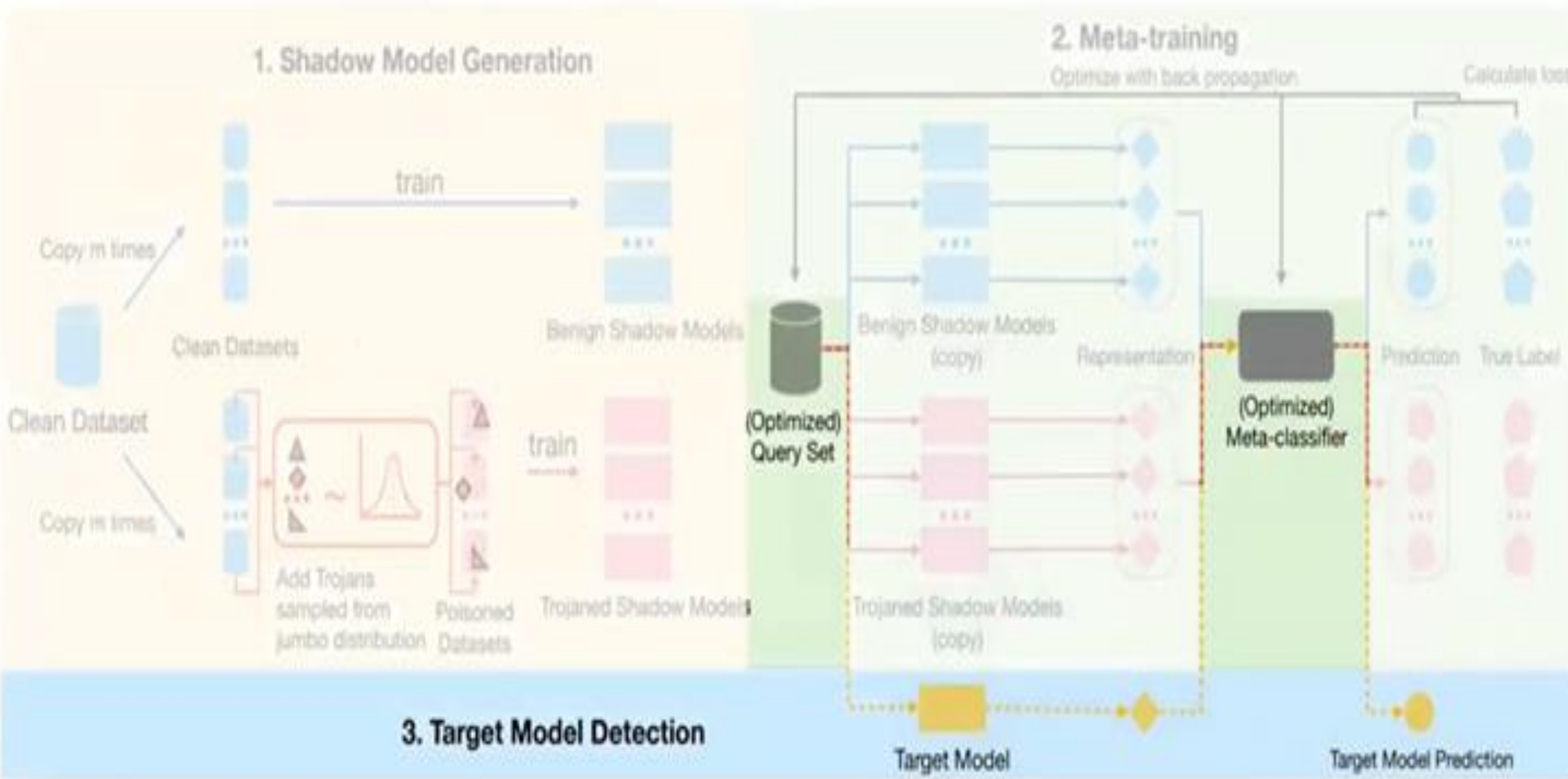
- One-class SVM: fit an SVM that separates between all training data and the origin.
  - In practice we use the one-class neural network technique [1] to train the meta-classifier.



[1] Anomaly detection using one-class neural networks, Chalapathy et al. 2018.

Step 3: Target Model Detection

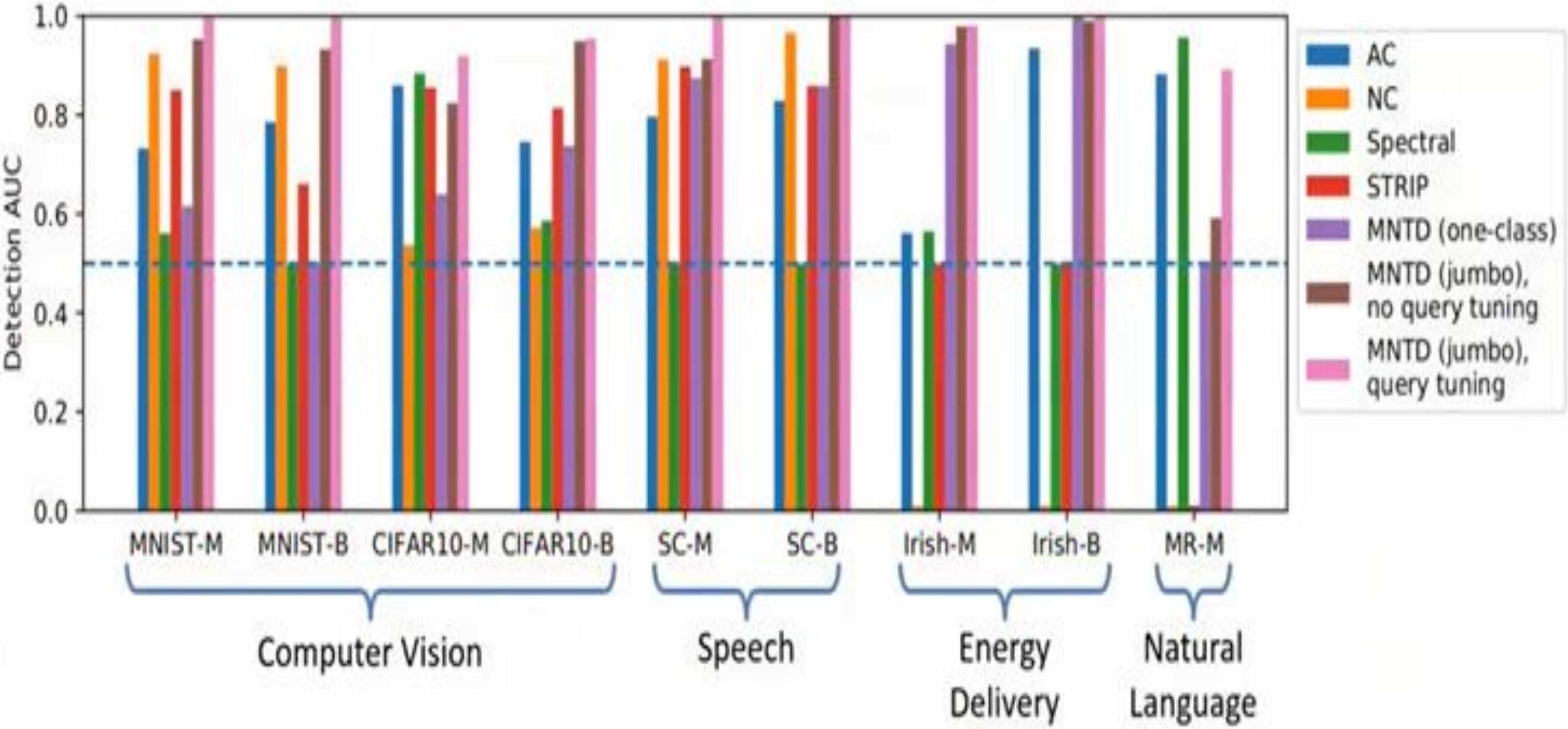
Seyed saeid vakil
s.s.vakil64@gmail.com



Detection Results

Seyed saeid vakil

s.s.vakil64@gmail.com



[AC] Detecting backdoor attacks on deep neural networks by activation clustering, Chen et al. 2018.

[NC] Neural cleanse: Identifying and mitigating backdoor attacks in neural networks, Wang el al. S&P 2019.

[Spectral] Spectral signatures in backdoor attacks, Tran el al. NeurIPS 2018.

[STRIP] STRIP: A Defence Against Trojan Attacks on Deep Neural Networks, Gao et al. 2019

Out-of-distribution Patterns

Seyed saeid vakil

s.s.vakil64@gmail.com

Trojan Shape	MNIST			CIFAR-10		
	Pattern Mask	Trojaned Example	Detection AUC	Pattern mask	Trojaned Example	Detection AUC
Apple			96.73%			89.38%
Corners			98.74%			93.09%
Diagonal			99.80%			97.57%
Heart			99.01%			93.82%
Watermark			99.93%			97.32%

Different Types of Trojan Attacks

Seyed saeid vakil

s.s.vakil64@gmail.com

- Parameter attack (denoted by -P)
  - Generate a trigger pattern, then fine-tune the model parameters to make it Trojaned.

- Latent attack (denoted by -L)
  - Generate a trigger pattern, then train a malicious model which does not contain a Trojan until it is fine-tuned by the consumer.

Different Types of Trojan Attacks

Seyed saeid vakil
s.s.vakil64@gmail.com

- Parameter attack (denoted by -P)
  - Generate a trigger pattern, then fine-tune the model parameters to make it Trojaned.

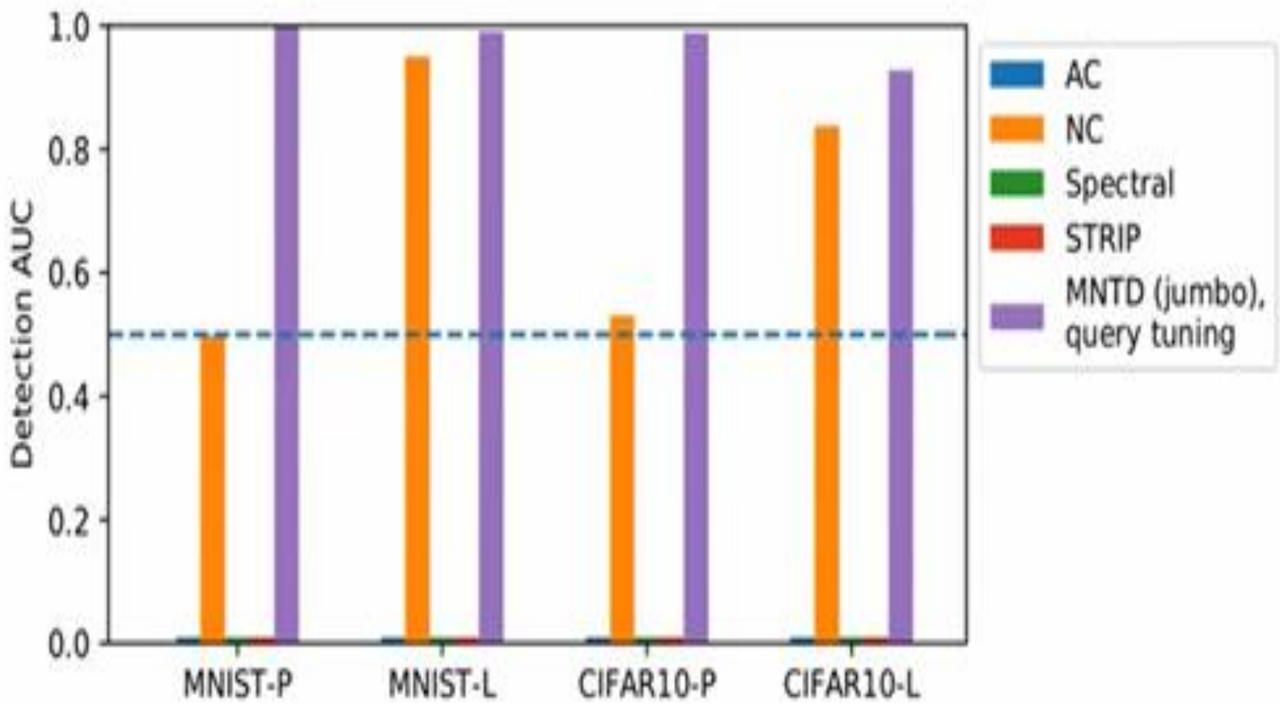
- Latent attack (denoted by -L)
  - Generate a trigger pattern, then train a malicious model which does not contain a Trojan until it is fine-tuned by the consumer.

These two attacks do not poison the training dataset!

Detection Results – Unforeseen Attacks

Seyed saeid vakil

s.s.vakil64@gmail.com



Detection Results – Unforeseen Attack Goals

Seyed saeid vakil

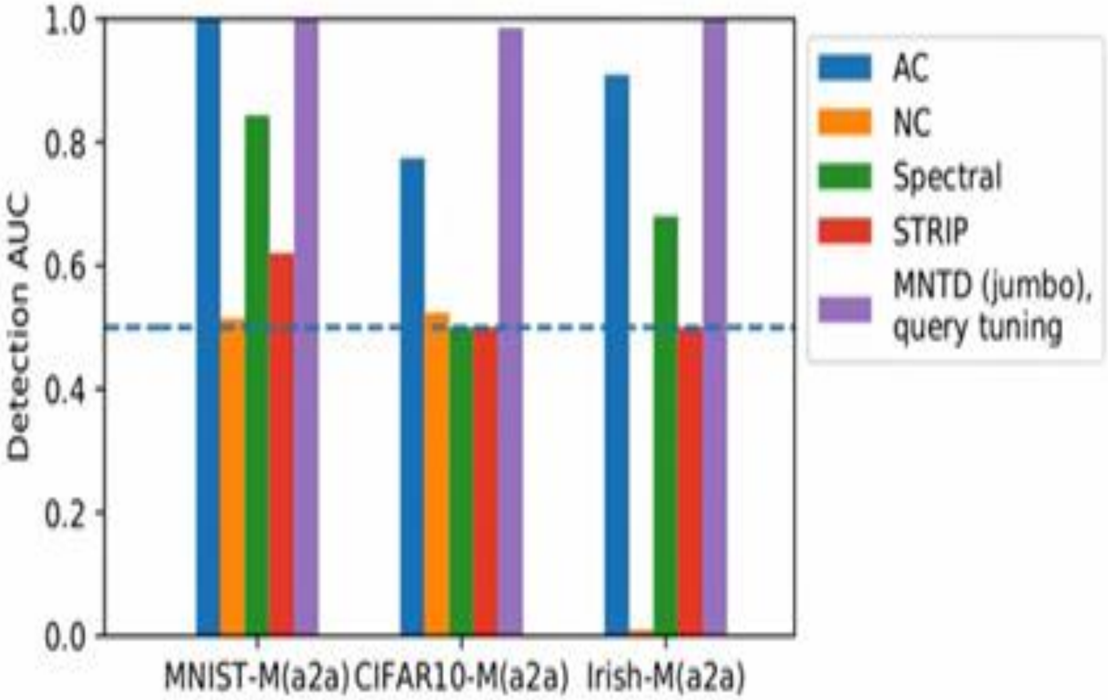
s.s.vakil64@gmail.com

- What we have used so far: Single-target attack goal
  - whenever the model sees the trigger pattern, it classifies the input as a **specific** class.

- Unforeseen attack goal: All-to-all attack goal
  - when the model sees the trigger pattern, it will **change** its prediction from the i-th class to the $((i+1)\%c)$-th class.
  - c is the number of classes.

Detection Results – Unforeseen Attacks

Seyed saeid vakil
s.s.vakil64@gmail.com



Time Efficiency of MNTD

Seyed saeid vakil

s.s.vakil64@gmail.com

- Offline Preparation: shadow model generation + meta-training.
- Inference: Target model detection.

Approach	Time (sec)
AC	27.13
NC	57.21
Spectral	42.55
STRIP	738.5
MNTD	$\mathbf{2.629 \times 10^{-3}}$
MNTD (offline preparation time)	$\sim 4096 \times 12 + 125$

Defend against Adaptive Attack

Seyed saeid vakil

s.s.vakil64@gmail.com

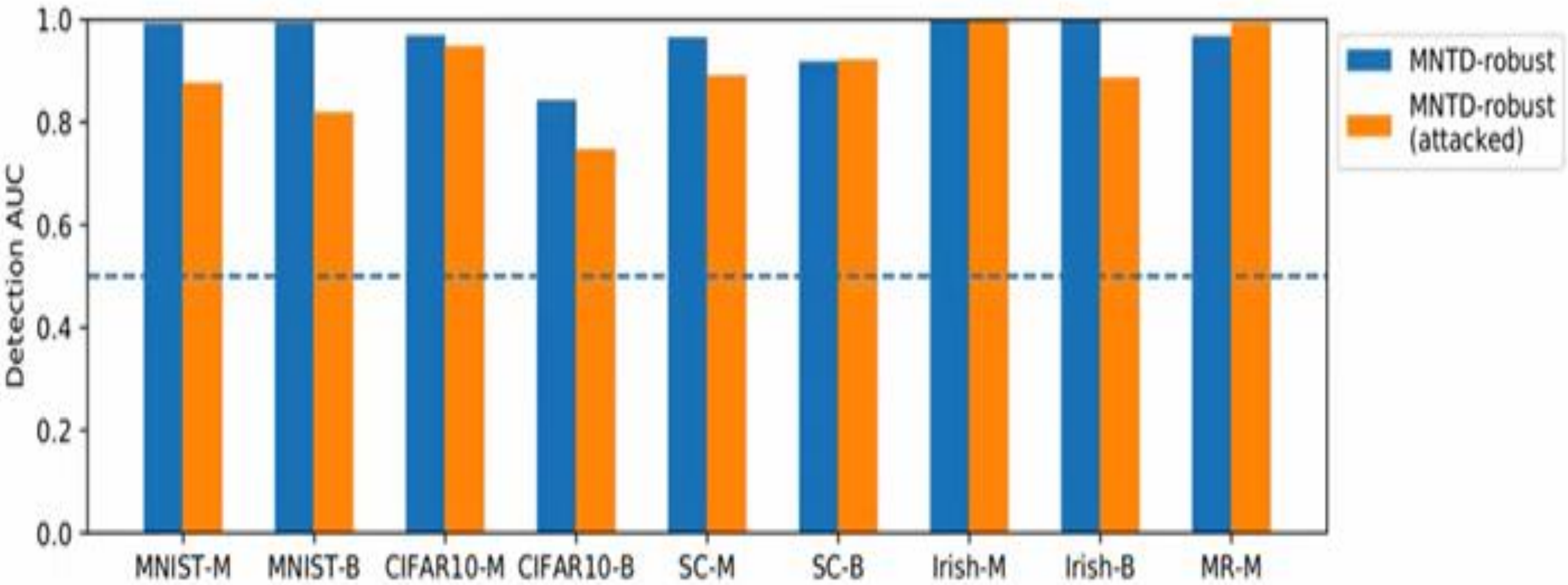
- What if the attacker knows our model and algorithm?
  - The attacker can intentionally generate Trojaned models that seem benign to our detection model.
  - Exp results: the attacker can evade the detection with >99% probability.

- Solution: Incorporate randomness in the algorithm!

- Robust Meta Neural Trojan Detection
  - At running time, randomly sample a meta-classifier and keep it unchanged.
  - Fine-tune the query set w.r.t. the random classifier.
  - Use the fine-tuned query set and random classifier to detect the Trojan.

Detection Results against Adaptive Attack

Seyed saeid vakil
s.s.vakil64@gmail.com



Take-away points

Seyed saeid vakil

s.s.vakil64@gmail.com

- Meta Neural Analysis achieves a good performance in the detection of Trojaned ML models.
  - It also generalizes well to unforeseen Trojans.
- The inference of MNTD is very efficient.
  - Although it takes a long time to train the MNTD model.
- By incorporating randomness, we can detect Trojans even when the attacker knows our detection algorithm.