GitHub Repo

Test-Time Poisoning Attacks Against Test-time Adaptation Models

Tianshuo Cong

Tsinghua University

congtianshuo@tsinghua.edu.cn

Xinlei He

Hong Kong University of Science and Technology (Guangzhou)

xinleihe@hkust-gz.edu.cn

Yun Shen

NetApp

yun.shen@netapp.com

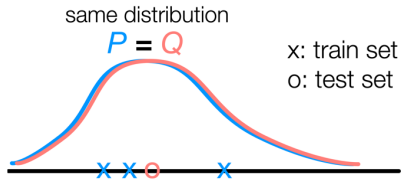
Yang Zhang

CISPA Helmholtz Center for Information Security

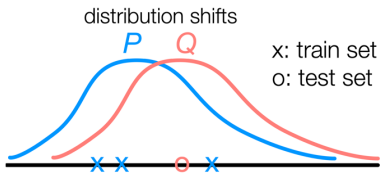
zhang@cispa.de

Background

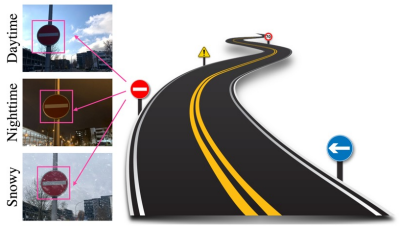
- **Deploying Deep Learning (DL) Models In The Wild**
  - Nowadays, DL has achieved remarkable performance.
  - Deploying DL models in the real-world poses a significant challenge due to distribution shift.
- **What Is Distribution Shift?**
  - DL models are usually trained and tested on the same distribution of data.
  - During inference, the parameters of the model are fixed.
  - Distribution shift occurs when the training and test datasets come from different distributions.



Theoretical [1]



Practical



(a) Single Recognition

(b) Multiple Recognition [2]

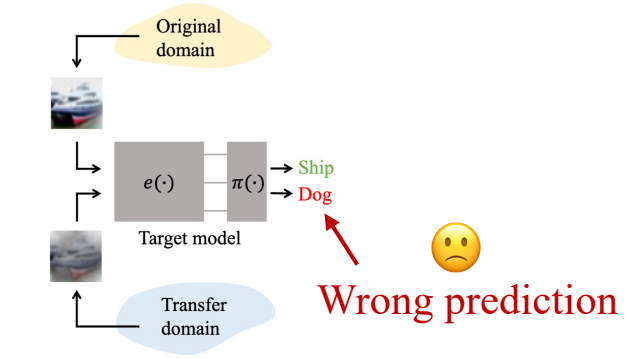
Fig. DL-based traffic sign recognition in the changeable weather scene.

[1] https://yueatsprograms.github.io/ttt/home.html.

[2] M. Jehanzeb Mirza, et al. The Norm Must Go On: Dynamic Unsupervised Domain Adaptation by Normalization. CVPR 2022.

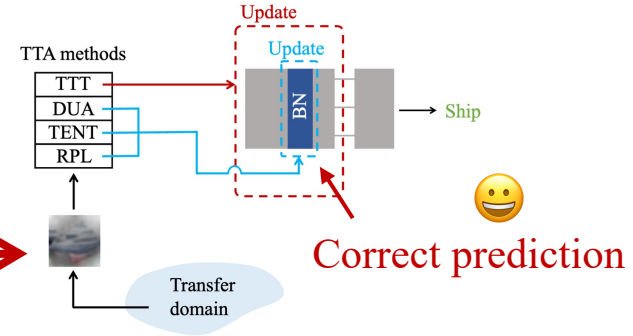
Background

- **How To Tackle Distribution Shift?**
  - Prior approaches to enhance DL model's generalization focused on the training process.
  - Learn more distribution types in advance.
  - 😕 Cannot be applicable to the diverse and unseen distribution.



(a) Inference w/o TTA.

- **Test-Time Adaptation (TTA)**
  - TTA is an emerging technique to tackle distribution shifts.
  - TTA has been leveraged in several real-world security-sensitive scenarios, such as autonomous driving, medical diagnosis, etc.
  - The distribution information contained in the test data can help the model to adjust itself.
  - The prediction will be made after updating the model via TTA.



(b) Inference w/ TTA.

Motivation & Threat Model

- **Our Motivation**
  - Though proven successful in improving the generalization of ML models, TTA paradigms may introduce a new attack surface for adversaries.
  - The parameters of the target model can be fine-tuned with potential malicious samples at test time.
  - We propose the first test-time poisoning attacks (TePA) against TTA models.

- **Threat Model**
  - **Adversary's Goal:** Degrade the target model's performance by nudging the model in a "wrong direction" by feeding poisoned samples at test time.
  - **Adversary's Knowledge:**
    - ✓ Know which TTA method the target model uses.
    - ✓ Can collect a surrogate model to generate poisoned samples.
    - ✓ Cannot intervene the training process of the target model
    - ✓ Do not have access to the model parameters of the target model at any time
  - **Attack Scenario:** benign samples uploaded by legitimate users and the poisoned samples fed by the adversaries are in the same pipeline.

Attack Challenges

- **Traditional Poisoning Attacks**
  - The training set is maliciously modified to degrade model performance

$$\max_{\mathcal{A}} \mathcal{L}(\mathcal{D}; \theta^*) \quad \text{where} \quad \theta^* = \text{argmin}_\theta \mathcal{L}(\mathcal{A}(\mathcal{D}_{train}); \theta)$$

  - Common method: mismatched "sample-label pairs"

- **Compared with Training-time, for test-time poisoning:**
  - Attackers can only feed unlabeled test data
  - Test data is usually used only once to update model parameters
  - The updated parameters of the model may be only partial

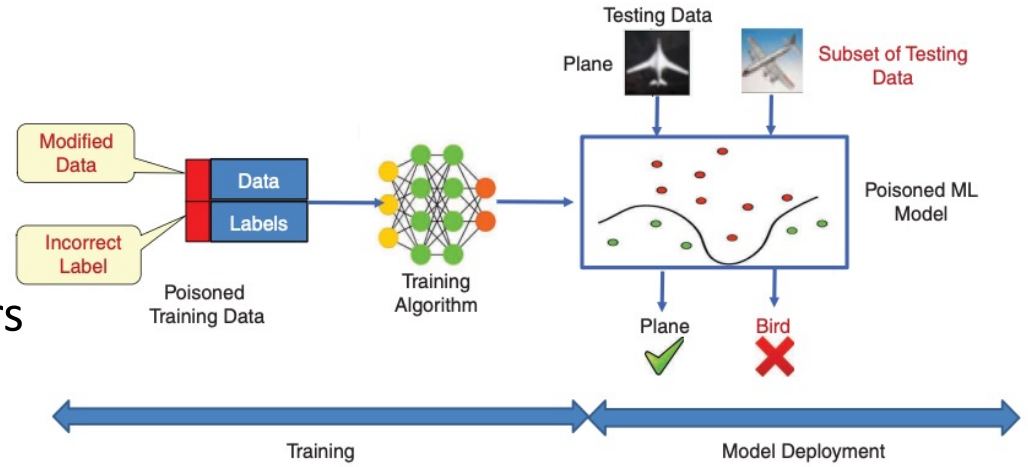


Fig. Training-time Poisoning Attacks.[3]

[3] Alina Oprea, et al. Poisoning Attacks Against Machine Learning: Can Machine Learning Be Trustworthy? Computer, 2022, 55(11): 94-99.

TTA Method-1: TTT

- **Test-Time Training** (ICML'20)[4]

- **Training Process**
  - Y-structured NN: $e(x;\theta_e), \pi_m(x;\theta_m), \pi_s(x;\theta_s)$
  - Multi-task learning:

$$\min_{e,\pi_s,\pi_m} \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}_m(x_i, y_i; e, \pi_m) + \mathcal{L}_s(x_i; e, \pi_s)$$

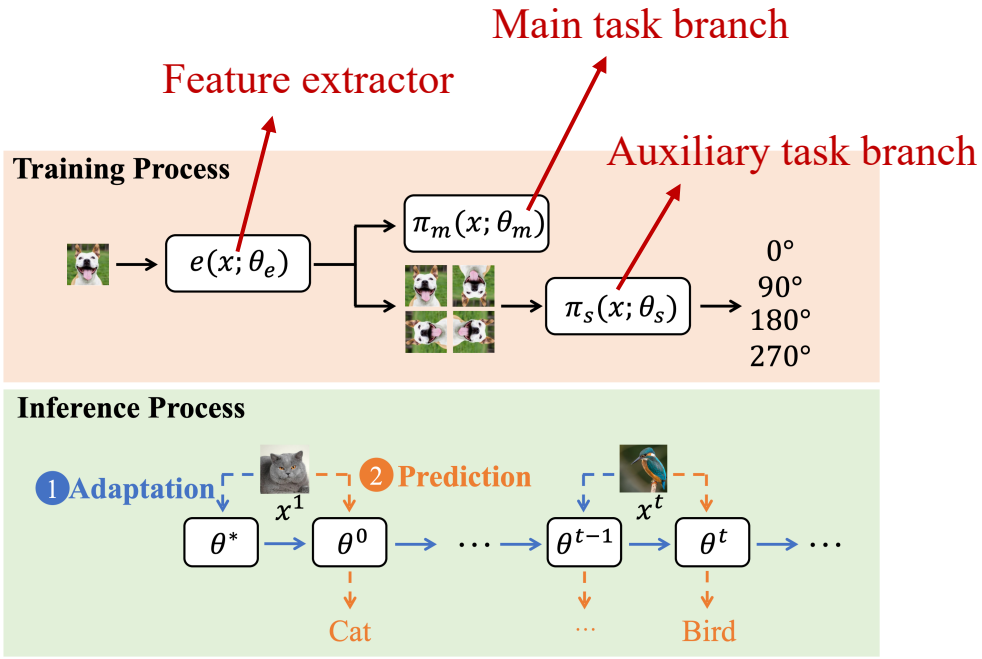


Fig. Overview of TTT.

- **Inference Process**
  - Test sample arrives one-by-one.
  - Initialization $(t=0)$: $\theta_0 = (e^*, \pi^*)$.
  - When $t=1, e^1, \pi_s^1 = \min_{e^*,\pi_s^*} \mathcal{L}_s(x^0; e^*, \pi_s^*)$, the prediction is $\hat{y}^0 = \pi_m\left(e^1(x^0)\right)$.
  - The parameter at time t is $\theta_t = (e^t, \pi_s^t)$, and the parameter used to inference is $\pi_m \circ e^{t+1}$.

[4] Yu Sun, et al. Test-Time Training with Self-Supervision for Generalization under Distribution Shifts. ICML 2020.

TTA Method-2: TENT

- **TENT: Test Entropy Minimization** (ICLR 2021)[5]

- **Inference Process**
  - Test-time normalization + Entropy minimization.
  - Test samples arrive batch-by-batch.
  - BN layer: $BN(x;\mu_s,\sigma_s,\gamma_s,\beta_s)=\frac{x-\mu_s}{\sqrt{\sigma_s+\epsilon}}\cdot\gamma_s+\beta_s,$

    where $\mu_s=\mathbb{E}[\mathcal{D}_s]$, $\sigma_s=Var[\mathcal{D}_s]$.
  - TENT updates BN layer as

$$\gamma_t \leftarrow \gamma_{t-1}-\partial\mathcal{L}_{tent}/\partial\gamma_{t-1},$$
$$\beta_t \leftarrow \beta_{t-1}-\partial\mathcal{L}_{tent}/\partial\beta_{t-1},$$

    where $(\gamma_0,\beta_0)=(\gamma_s,\beta_s)$ and

$$\mathcal{L}_{tent}(f(\mathbf{x}^t))=-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}p(j|x_i^t)\log p(j|x_i^t).$$

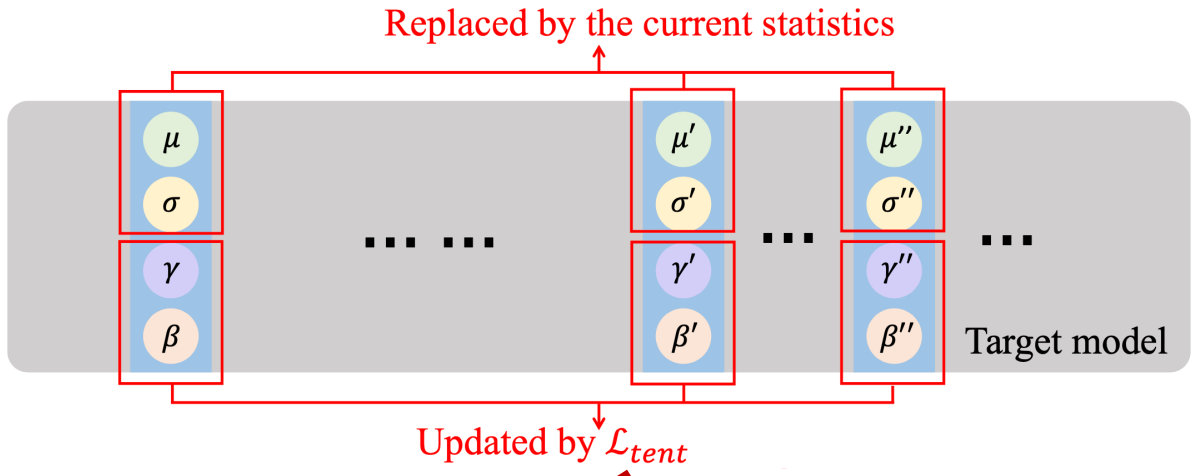


Fig. Overview of TENT.

[5] Dequan Wang, et al. Tent: Fully Test-time Adaptation by Entropy Minimization. ICLR 2021.

TTA Method-3: RPL

- **Robust Pseudo-Labeling** (TMLR'22)[6]

- **Inference Process**
  - The only different setting to TENT is the loss function.
  - RPL updates BN layer:

$$\gamma_t \leftarrow \gamma_{t-1} - \partial \mathcal{L}_{rpl}(f(x^t))/\partial \gamma_{t-1},$$

$$\beta_t \leftarrow \beta_{t-1} - \partial \mathcal{L}_{rpl}(f(x^t))/\partial \beta_{t-1},$$

where $q \in (0, 1]$,

$$\mathcal{L}_{rpl}(f(x^t)) = \frac{1}{N}\sum_{i=1}^{N} q^{-1}(1 - p(\Psi|x_i^t)^q),$$

and

$$\Psi = \arg\max_{j=1,\ldots,k} p(j|x_i^t).$$

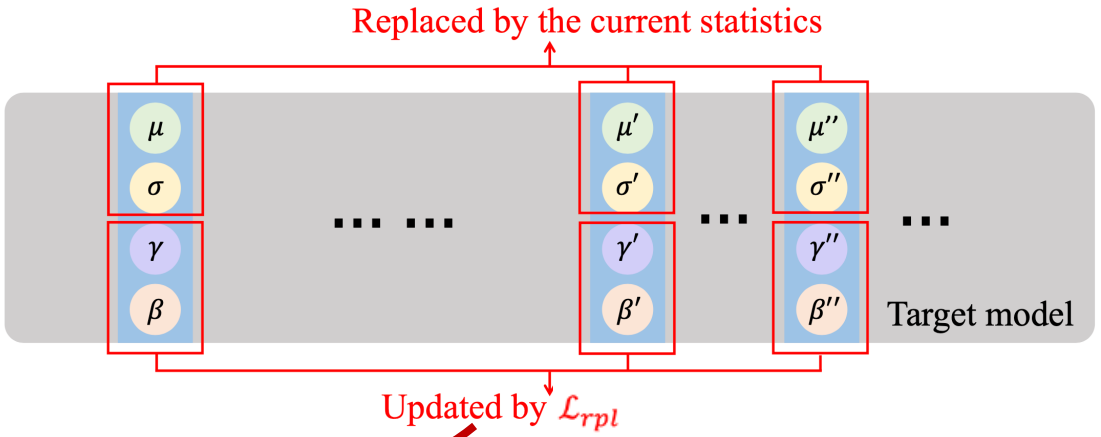


Fig. Overview of RPL.

[6] Evgenia Rusak, et al. If your data distribution shifts, use self-learning. TMLR 2022.

TTA Method-4: DUA

- **Dynamic Unsupervised Domain Adaption** (CVPR'22)[7]

- **Training Process**
  - BN layer is updated as

$$\mu_k \leftarrow (1-\rho)\cdot\mu_{k-1}+\rho\cdot\mu_k$$
$$\sigma_k^2 \leftarrow (1-\rho)\cdot\sigma_{k-1}^2+\rho\cdot\sigma_k^2$$

- **Inference Process**
  - Test sample arrives one-by-one.
  - The single sample is augmented to form a small batch.
  - BN layer keeps being updated as

$$\hat{\mu}_t = (1-(\rho_t+\xi))\cdot\hat{\mu}_{t-1}+(\rho_t+\xi)\cdot\mu_t,$$
$$\hat{\sigma}_t^2 = (1-(\rho_t+\xi))\cdot\hat{\sigma}_{t-1}^2+(\rho_t+\xi)\cdot\sigma_t^2,$$

where $\mu_0=\mu_s, \sigma_0^2=\sigma_s^2,\ \ \rho_k=\rho_{k-1}\cdot\omega,\ \rho_k=0.1, \omega\in(0,1),\ 0<\zeta<\rho_0.$

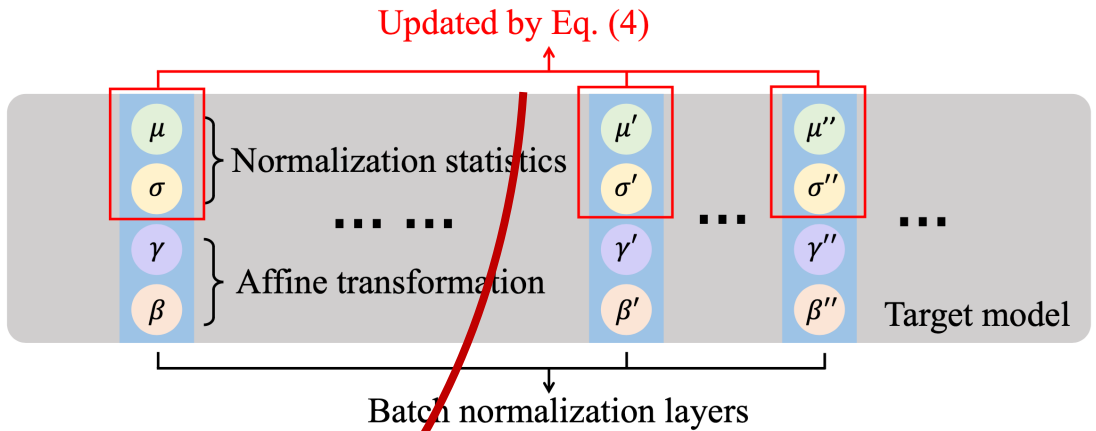


Fig. Overview of DUA.

[7] M. Jehanzeb Mirza, et al. The Norm Must Go On: Dynamic Unsupervised Domain Adaptation by Normalization. CVPR 2022.

TTA Method: Summary

- Four TTA methods discussed in our paper

Table. Statistical Information

TTA Method	Parameters to adjust	Test data stream	Venue
TTT [45]	Feature extractor	Point-by-point	ICML 2020
DUA [32]	BN Layers	Point-by-point	CVPR 2022
TENT [50]	BN Layers	Batch-by-batch	ICLR 2021
RPL [38]	BN Layers	Batch-by-batch	TMLR 2022

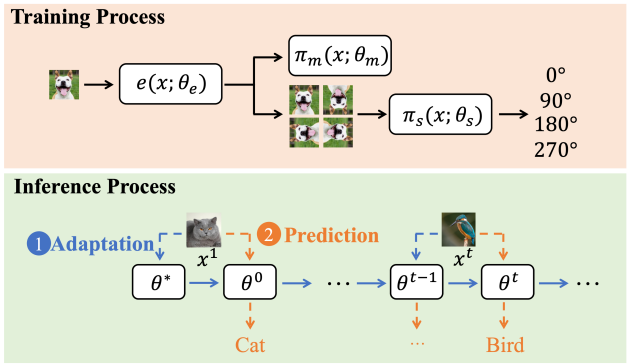


Fig. Overview of TTT.

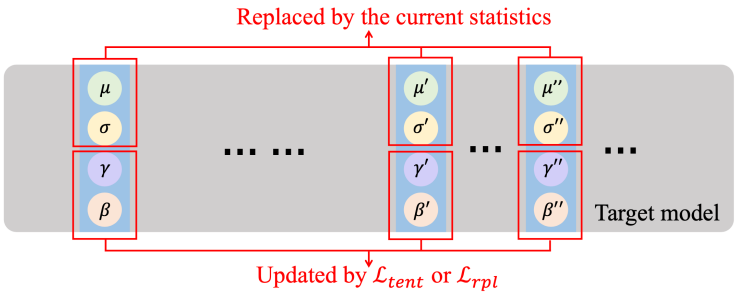


Fig. Overview of TENT and RPL.

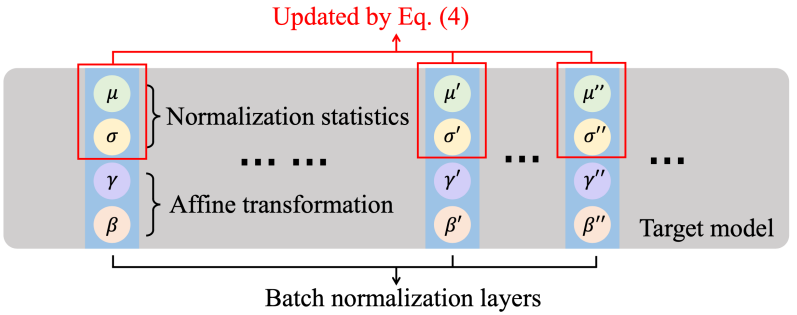


Fig. Overview of DUA.

Methodology (Let's poison TTA-models!)

- **Attack Pipeline**
  - Surrogate model training
  - Poisoned sample generation
  - Target model poisoning

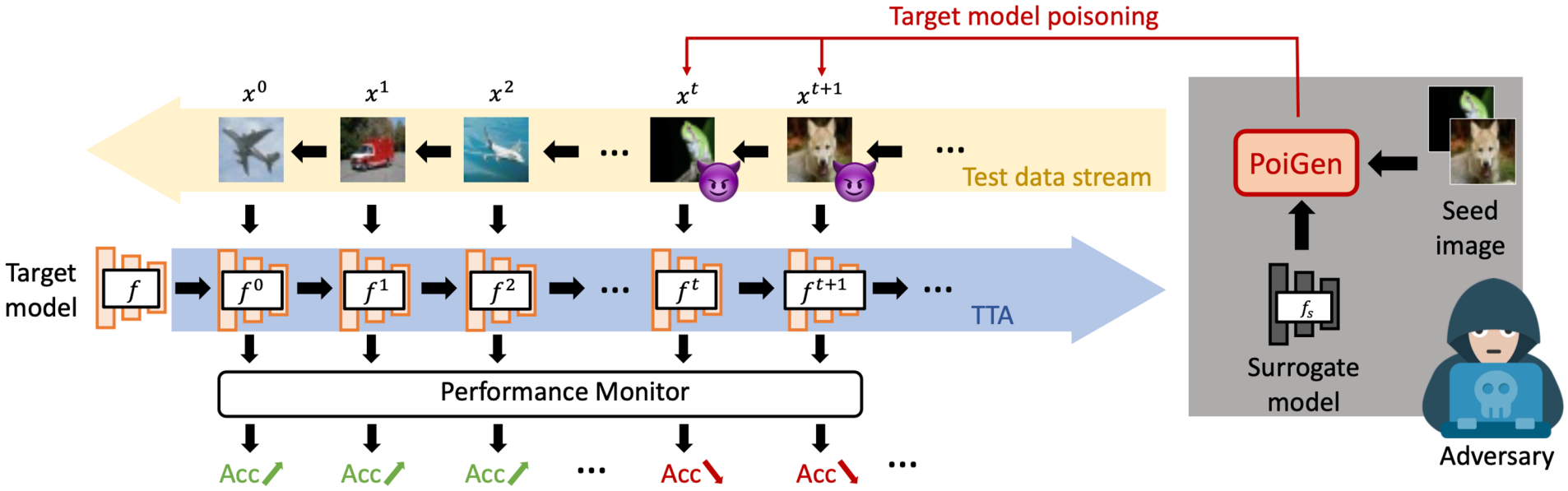
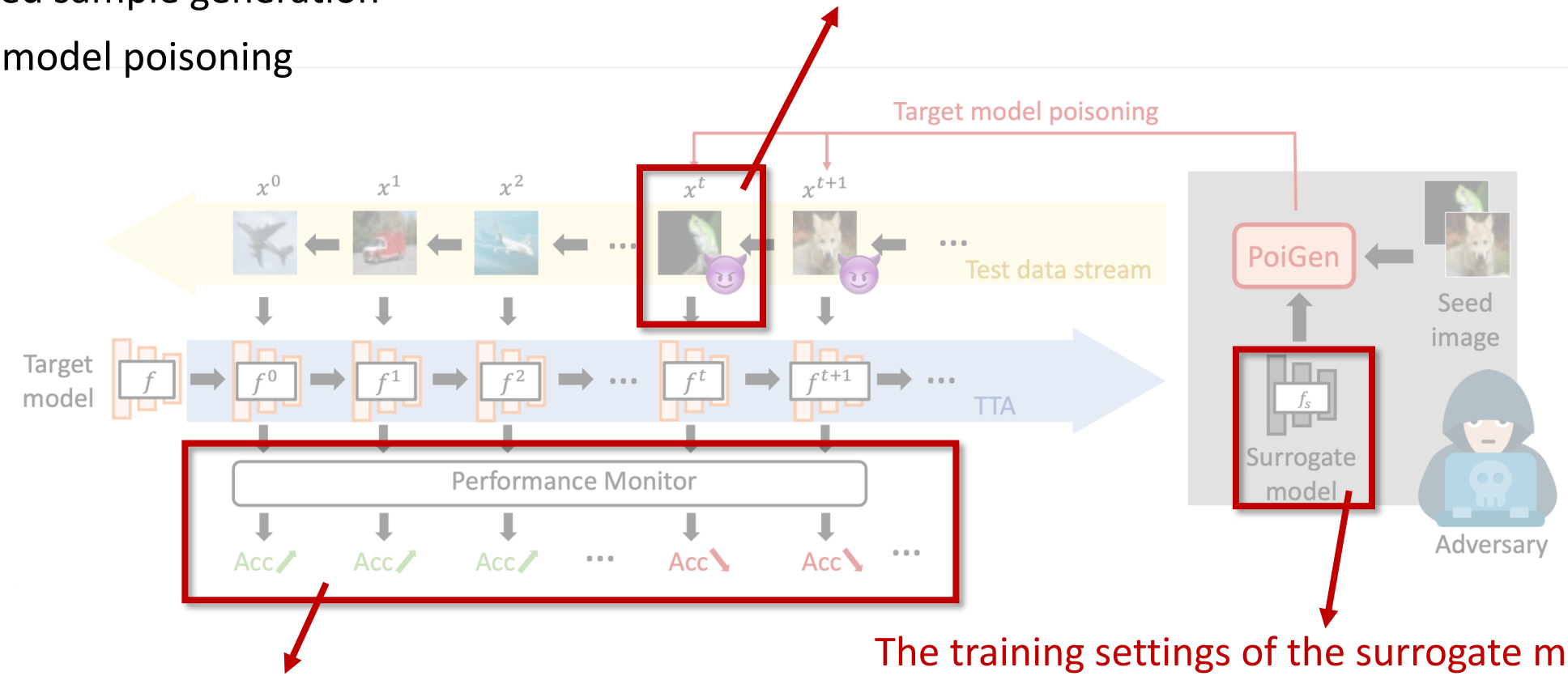


Fig. Workflow of our test-time poisoning attacks against TTA-models.

Methodology (Let's poison TTA-models!)

- **Attack Pipeline**
  - Surrogate model training
  - Poisoned sample generation
  - Target model poisoning

The poisoned samples are generated based on the self-supervised learning task loss within the TTA methods (*gradient ascent direction*).



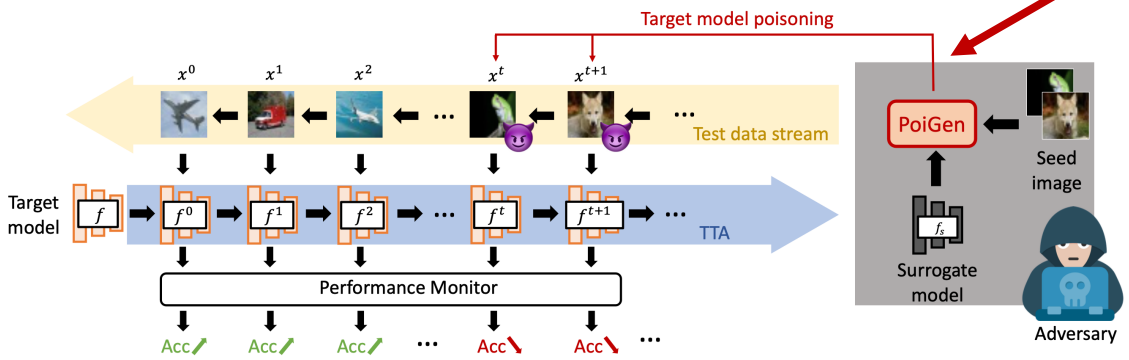
We use a fixed evaluation dataset to monitor the changes in model performance.

The training settings of the surrogate model are different from those of the target model.

Methodology

- **Attack Pipeline**
  - Surrogate model training
  - Poisoned sample generation
  - Target model poisoning

Use DIM to enhance the transferability of the poisoned samples.



Algorithm 1: PoiGen

Input: Seed image x_{in}, surrogate model f_s, the target TTA method $\mathcal{A}$, loss function $\mathcal{L}_{poi}$, the perturbation budget ϵ, updating step α;

Output: Poisoned sample x';

Def DIM$(x,\ y,\ f,\ L,\ \epsilon)$**:**
- $g = 0$;
- $\mu = 1$;
- $p = 0.5$;
- **for** $j = 1$ to I_{adv} **do**
  - $x = T(x, p)$;
  - **if** y is not None **then**
    - $g = \mu \cdot g + \frac{\nabla_{x_{in}} L(f(x), y)}{||\nabla_{x_{in}} L(f(x), y)||_1}$;
  - **else**
    - $g = \mu \cdot g + \frac{\nabla_{x_{in}} L(f(x))}{||\nabla_{x_{in}} L(f(x))||_1}$;
  - $x^{adv} = x_{in} + \alpha \cdot \text{sign}(g)$;
  - $\delta = \text{Clip}(x^{adv} - x_{in}; -\epsilon, +\epsilon)$;
  - $x = \text{Clip}(x_{in} + \delta; 0, 1)$;
- **return** x.

Main function PoiGen $(\mathcal{A},\ x_{in},\ f_s,\ \mathcal{L}_{poi},\ \epsilon)$**:**
- **if** $\mathcal{A}$ is TTT **then**
  - $x' = x_{in}$;
  - **for** $i = 1$ to I_{iter} **do**
    - **for** y' = 1 to 4 **do**
      - $x_{rot} = \text{rot90}(x', y')$;
      - $x' = \text{DIM}(x_{rot}, y', f_s, \mathcal{L}_{poi}, \epsilon)$;
- **else if** $\mathcal{A}$ is TENT or RPL **then**
  - $x' = \text{DIM}(x_{in}, y = \text{None}, f_s, \mathcal{L}_{poi}, \epsilon)$;
- **else if** $\mathcal{A}$ is DUA **then**
  - $x' = x + \epsilon \cdot \mathcal{N}(\mu, \sigma^2)$ (See Equation 13);
- **return** x'.

Use rotation prediction loss to poison TTT-models

Use $\mathcal{L}_{tent}$ or $\mathcal{L}_{rpl}$ to poison TENT-models or RPL-models

Gaussian noise is enough to poison DUA-models

Evaluation: Frozen Target Model

- **The Utility of The Frozen Target Model**
  - DNNs cannot be robust enough on distribution shifts.
  - Y-structured DNNs are more robust than naïve DNNs.

TABLE 1: The utility of the frozen target model (%).

Dataset	Target Model	Acc			
		Ori	Gls-5	Fog-5	Con-5
CIFAR-10	C10-Res18@Y4	93.70	61.90	71.40	83.60
	C10-Res50@Y4	92.80	56.60	68.00	78.50
	C10-Res18	93.00	58.10	64.80	19.20
	C10-Res50	94.20	62.60	70.80	24.90
CIFAR-100	C100-Res18@Y3	71.40	20.90	41.40	48.70
	C100-Res50@Y3	65.20	24.70	31.40	30.80
	C100-Res18	73.50	24.60	32.60	11.50
	C100-Res50	76.20	25.50	38.30	12.30

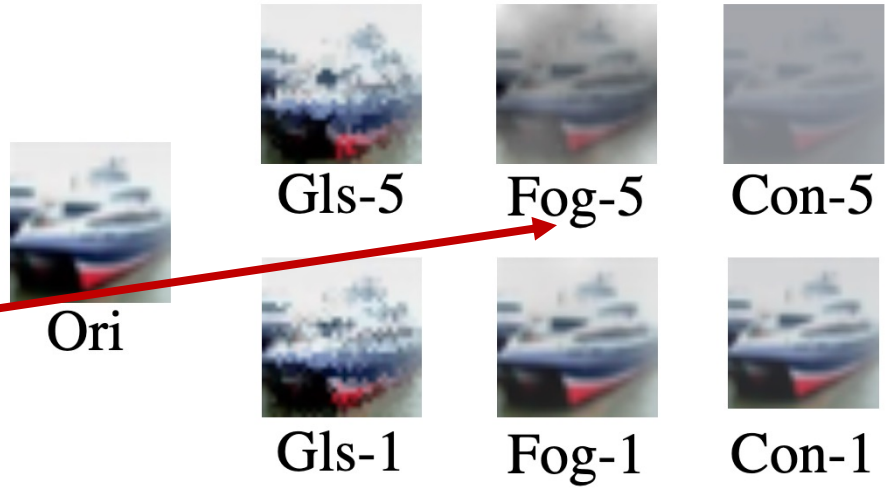


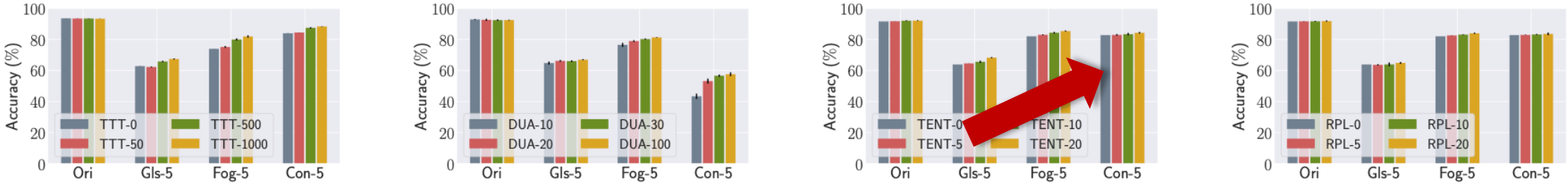
Fig. The corrupted samples from CIFAR-10-C.

😕 Serious performance degradation on corrupted test samples.

Evaluation: TTA-Models

- **The Utility of TTA Methods**
  - The performance of the target models can be improved by the TTA methods.
  - TENT and RPL both have a greater ability to enhance the model performance.
  - TENT can achieve better performance than RPL.

😀 "As the amount of benign samples increases, the model gains more performance improvement."



(a) TTT (b) DUA (c) TENT (d) RPL

Figure 4: Utility of TTA methods. The target model is ResNet-18 trained on CIFAR-10. The x-axis represents different evaluation datasets. The y-axis represents the prediction accuracy.

Evaluation: Poisoning TTA-Models

- **TePA Against TTA Models**
  - Regardless of the network architecture or the training dataset, our poisoned samples lead to a significant reduction in the prediction abilities of the target models.
  - Though the surrogate model has a different architecture and is trained on a different surrogate dataset, TePAs are still effective.

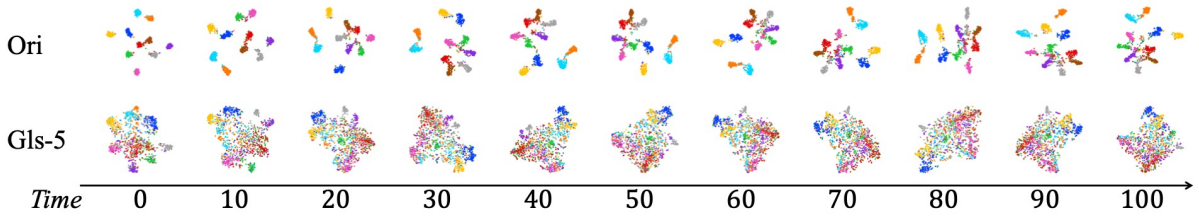
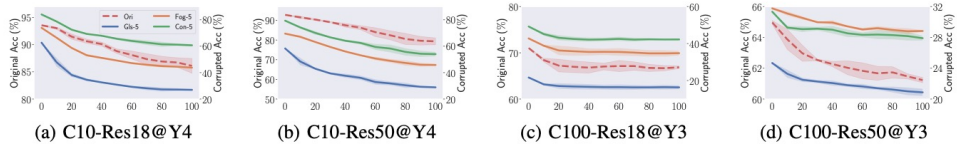
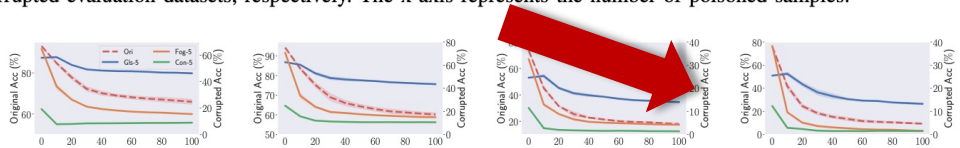


Fig. t-SNE visualization.



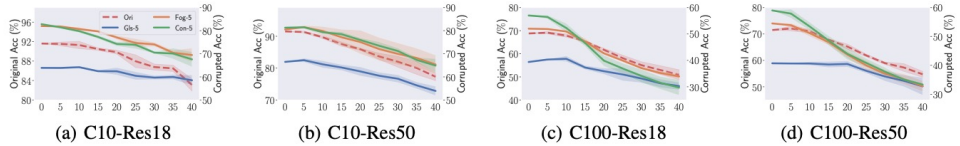
(a) C10-Res18@Y4 (b) C10-Res50@Y4 (c) C100-Res18@Y3 (d) C100-Res50@Y3

Figure 5: TePAs Against TTT-models. The left y-axis and the right y-axis represent the prediction accuracy on the original and corrupted evaluation datasets, respectively. The x-axis represents the number of poisoned samples.



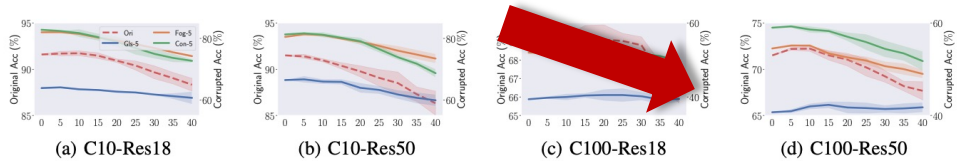
(a) C10-Res18 (b) C10-Res50 (c) C100-Res18 (d) C100-Res50

Figure 6: TePAs Against DUA-models. The left y-axis and the right y-axis represent the prediction accuracy on the original and corrupted evaluation datasets, respectively. The x-axis represents the number of poisoned samples.



(a) C10-Res18 (b) C10-Res50 (c) C100-Res18 (d) C100-Res50

Figure 7: TePAs Against TENT-models. The left y-axis and the right y-axis represent the prediction accuracy on the original and corrupted evaluation datasets, respectively. The x-axis represents the number of poisoned samples.

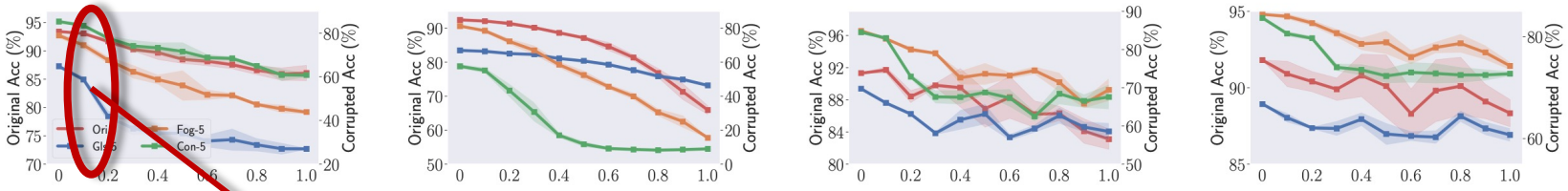


(a) C10-Res18 (b) C10-Res50 (c) C100-Res18 (d) C100-Res50

Figure 8: TePAs Against RPL-models. The left y-axis and the right y-axis represent the prediction accuracy on the original and corrupted evaluation datasets, respectively. The x-axis represents the number of poisoned samples.

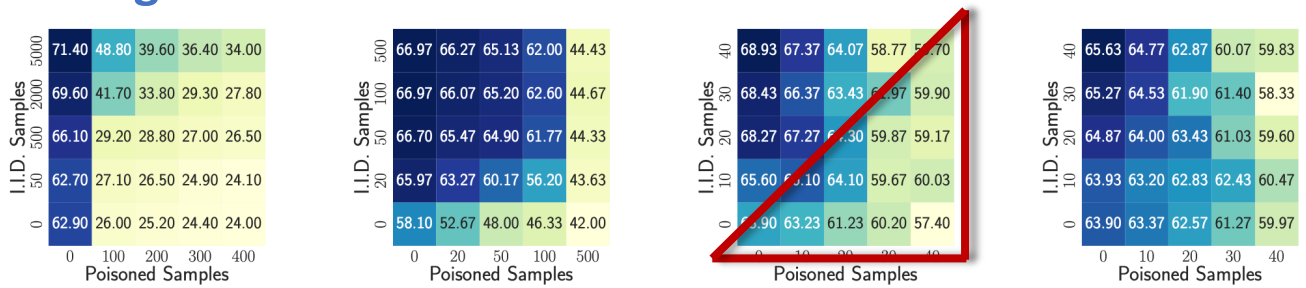
Evaluation: Poisoning Strategies

- **Uniformly Poisoning**



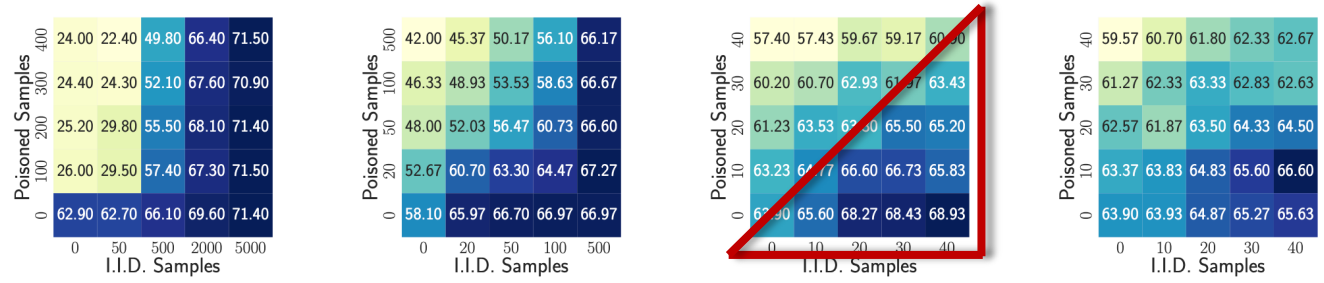
(a) TTT (b) DUA (c) TENT (d) RPL

- **Warming-up Before Poisoning**



(a) TTT (b) DUA (c) TENT (d) RPL

- **Warming-up After Poisoning**



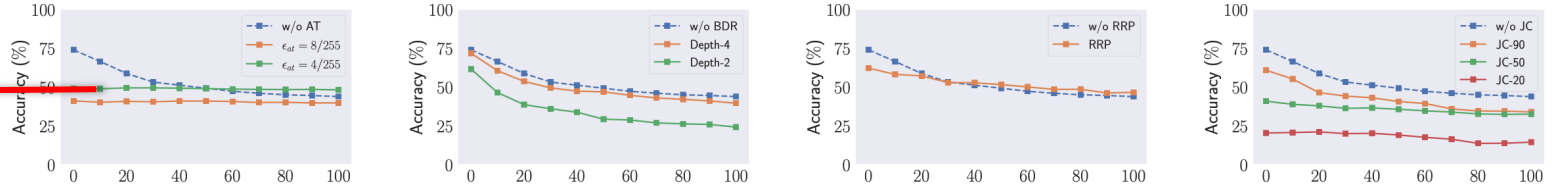
(a) TTT (b) DUA (c) TENT (d) RPL

Evaluation: Defenses

- **Four Potential Defenses**
  - Adversarial training (AT)
  - Bit-depth reduction (BDR)
  - Random resizing & padding (RRP)
  - JPEG compression (JC)

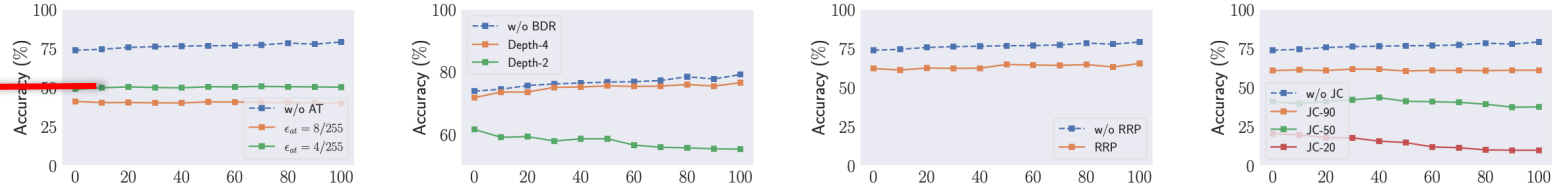
😕 "Poisoned samples can still degrade the target model's performance."

Poisoned samples



(a) AT (b) BDR (c) RRP (d) JC

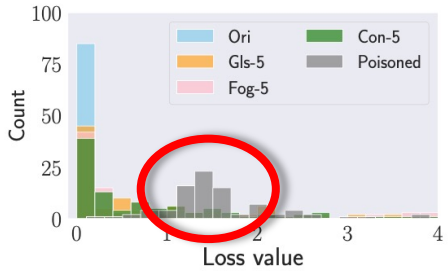
Benign samples



(a) AT (b) BDR (c) RRP (d) JC

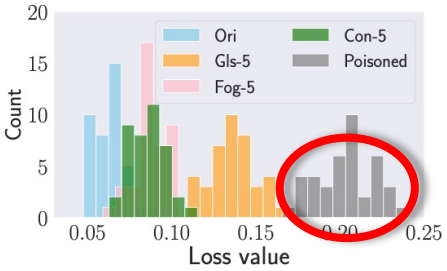
Discussion

- **The Statistics Results of The Loss Values**



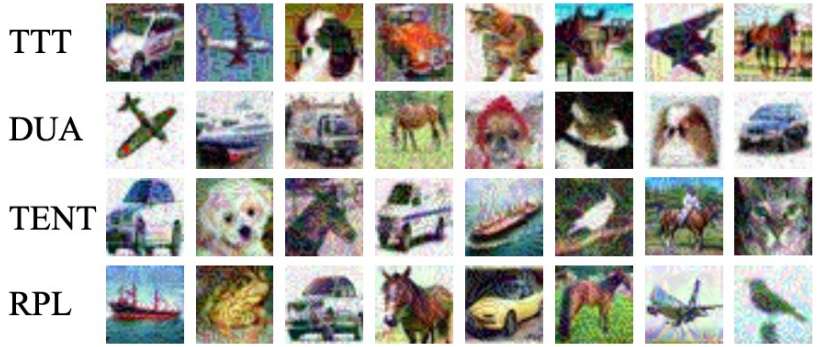
(a) TTT

(b) TENT



(c) RPL

- **Visualization Results of The Poisoned Samples**



Conclusion

- **Takeaways**
  - Empirical evaluations show that TePAs can successfully break the target TTA-models by degrading their performance to a large extent.
  - We notice that the recovery of the target model's performance is inevitable for our attacks

- **Future Work**
  - How to irreversibly degrade the target model's performance?
  - We advocate for the integration of defenses against test-time poisoning attacks into the design of future TTA methods

Thanks!

https://github.com/tianshuocong/TePA