CS 578: CYBER-SECURITY

PART V: TRUSTWORTHY ML

Sanghyun Hong
sanghyun.hong@oregonstate.edu





RECAP – WHERE WE ARE

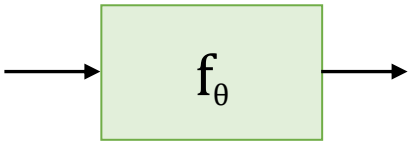
- Part I – Network/Internet security
  - The attacker is outside your machine
- Part II – Computer systems security
  - The attacker is inside your machine
  - Memory safety: how the attacker gets in
  - OS security: decide what the attacker may do (access control)
- Part III – Isolation and (software-induced) breaks
  - The attacker is inside your program or is the OS itself
  - The attacker exploits your hardware and breaks integrity and confidentiality
- Part IV – Software/Web security
  - The attacker turns bugs into exploits and delivers them through the web
- Part V – Trustworthy ML
  - The attacker targets ML models: their inputs and their training data (today!)

MACHINE LEARNING – A NEW KIND OF SOFTWARE

PRELIM – A CLASSIFIER

- Model f
  - Input x: an image
  - Output: a score for each class
  - Prediction: the class with the highest score
- Parameters θ
  - Millions of numbers inside f
  - Set by training, not written by a person





x

PRELIM – TRAINING

- Training data
  - Pairs (x, y): an image and its correct label
- Loss L(θ, x, y)
  - Small when f predicts y, large when it does not
- Training: change θ to lower the average loss
  - Gradient ∇θ L: which way to move θ
  - Repeat over small batches of data (SGD)
- Test
  - New inputs the model has never seen

$$\theta \leftarrow \theta - \eta \, \nabla_{\theta} \, L(\theta, x, y)$$

PRELIM – THE GRADIENT WITH RESPECT TO THE INPUT

- Training
  - Keep x fixed; move θ to lower the loss
- Attacker at test time
  - Keep θ fixed; move x to raise the loss
  - ∇x L: how the loss changes with each pixel
- Same tool, opposite goal

$$\theta \leftarrow \theta - \eta \nabla_{\theta} L(\theta, x, y)$$

$$x \leftarrow x + \eta \nabla_{x} L(\theta, x, y)$$

WHAT CAN THE ATTACKER DO TO AN ML MODEL?

- At test time: change the input
  - Adversarial examples (today)
- At training time: change the training data
  - Data poisoning
- After training: ask about the training data
  - Membership inference (today)

ADVERSARIAL EXAMPLES – FOOLING A TRAINED MODEL

ADVERSARIAL EXAMPLES

- A test-time input to a neural network
  - Crafted with the objective of fooling the network's decision(s)

ADVERSARIAL EXAMPLES – CONT'D

- A test-time input to a neural network
  - Crafted with the objective of fooling the network's decision(s)
  - That looks like a natural test-time input



Noisy test-time input

ADVERSARIAL EXAMPLES – CONT'D

- A test-time input to a neural network
  - Crafted with the objective of fooling the network's decision(s)
  - That looks like a natural test-time input



Prediction: **Panda**

Human-imperceptible Noise

Prediction: **Gibbon**

Goodfellow et al., *Explaining and Harnessing Adversarial Examples, International Conference on Learning Representations (ICLR)*, 2015.

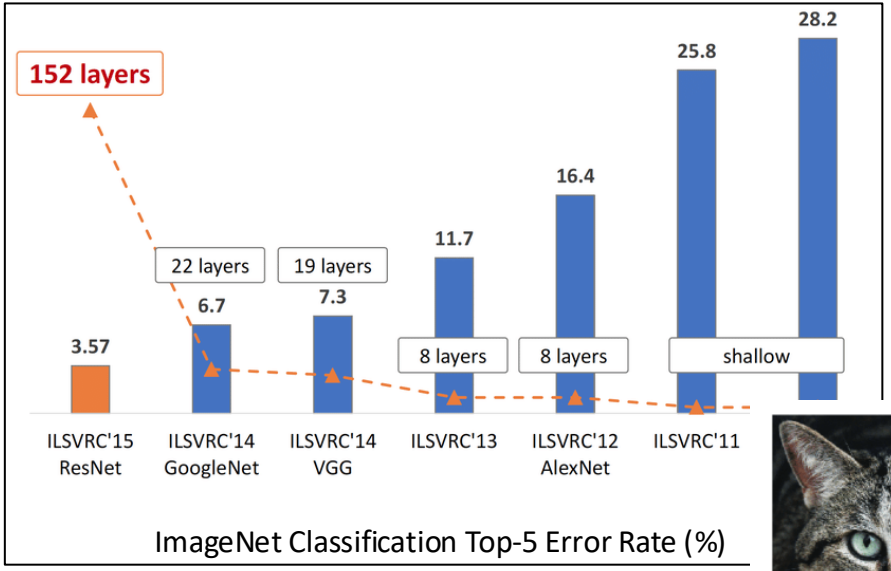
ADVERSARIAL EXAMPLES IN THE REAL WORLD

- from the security perspective: it makes ML-enabled systems unavailable



ADVERSARIAL EXAMPLES ARE COUNTER-INTUITIVE

- from the ML perspective: it is counter-intuitive





adversarial perturbation



88% tabby cat

99% guacamole

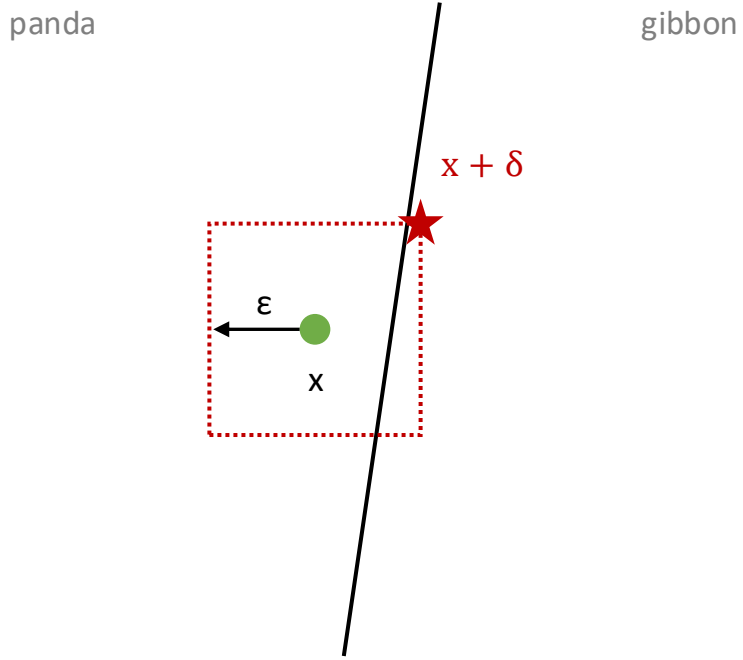
MADRY ET AL. – TRAINING MODELS ROBUST TO ADVERSARIAL EXAMPLES

MADRY ET AL. – RESEARCH QUESTION

- Question
  - Can we train networks that resist adversarial examples?
- Problem
  - Earlier defenses were broken by the next attack
  - What does robust mean, exactly?
- Approach
  - Define the attacker first
  - Write robustness as one optimization problem

MADRY ET AL. – THREAT MODEL

- Goal
  - Make the model misclassify an input
- Knowledge
  - White-box: the model and its parameters θ
- Capabilities
  - Change each pixel by at most ε
    - MNIST: ε = 0.3 (pixels in [0, 1])
    - CIFAR-10: ε = 8 (pixels in [0, 255])
  - Set of allowed changes: S



$$S = \{ \delta : |\delta_i| \leq \varepsilon \text{ for every pixel } i \}$$

MADRY ET AL. – SADDLE POINT PROBLEM

- Robust: low loss even on the worst input in S

$$\min_{\theta} \; \mathrm{E}_{(x,y)\sim D} \left[\max_{\delta \in S} L(\theta, x + \delta, y) \right]$$

Inner maximization (attacker):
find the worst δ in S

MADRY ET AL. – SADDLE POINT PROBLEM – CONT'D

- Robust: low loss even on the worst input in S

Outer minimization (defender):
find θ that keeps it low

$$\min_{\theta} \; \mathrm{E}_{(x,y)\sim D} \left[\; \max_{\delta \in S} L(\theta, x + \delta, y) \; \right]$$

Inner maximization (attacker):
find the worst δ in S

MADRY ET AL. – INNER MAXIMIZATION

- FGSM (Fast Gradient Sign Method)

$$x + \varepsilon \operatorname{sgn}(\nabla_x L(\theta, x, y)).$$

  - FGSM can be viewed as a simple one-step toward maximizing the loss (inner part)

MADRY ET AL. – INNER MAXIMIZATION

- FGSM (Fast Gradient Sign Method)

$$x + \varepsilon \operatorname{sgn}(\nabla_x L(\theta, x, y)).$$

  - FGSM can be viewed as a simple one-step toward maximizing the loss (inner part)

- PGD (Projected Gradient Descent)

$$x^{t+1} = \Pi_{x+\mathcal{S}} \left(x^t + \alpha \operatorname{sgn}(\nabla_x L(\theta, x, y)) \right).$$

FGSM

  - Multi-step adversary; much stronger than FGSM attack

MADRY ET AL. – OUTER MINIMIZATION

- PGD (Projected Gradient Descent)

$$x^{t+1} = \Pi_{x+\mathcal{S}} \left(x^t + \alpha \operatorname{sgn}(\nabla_x L(\theta, x, y)) \right).$$

  - Multi-step adversary; much stronger than FGSM attack

- Adversarial training
  - Make a model do correct prediction on adversarial examples
  - Training procedure
    - At each iteration of training
    - Craft PGD-t adversarial examples
    - Update the model towards making it correct on those adv examples

MADRY ET AL. – ADVERSARIAL TRAINING

- Robust training
  - Deep neural networks (DNNs) are universal function approximators[1]
  - DNNs may learn to be resistant to adversarial examples (a desirable function)
  - Adversarial training (AT):

Repeat:

1. Select minibatch B, initialize gradient vector $g := 0$
2. For each (x, y) in B:
   a. Find an attack perturbation $\delta^\star$ by (approximately) optimizing

$$\delta^\star = \underset{\|\delta\| \leq \epsilon}{\operatorname{argmax}} \ell(h_\theta(x + \delta), y)$$

   b. Add gradient at $\delta^\star$

$$g := g + \nabla_\theta \ell(h_\theta(x + \delta^\star), y)$$

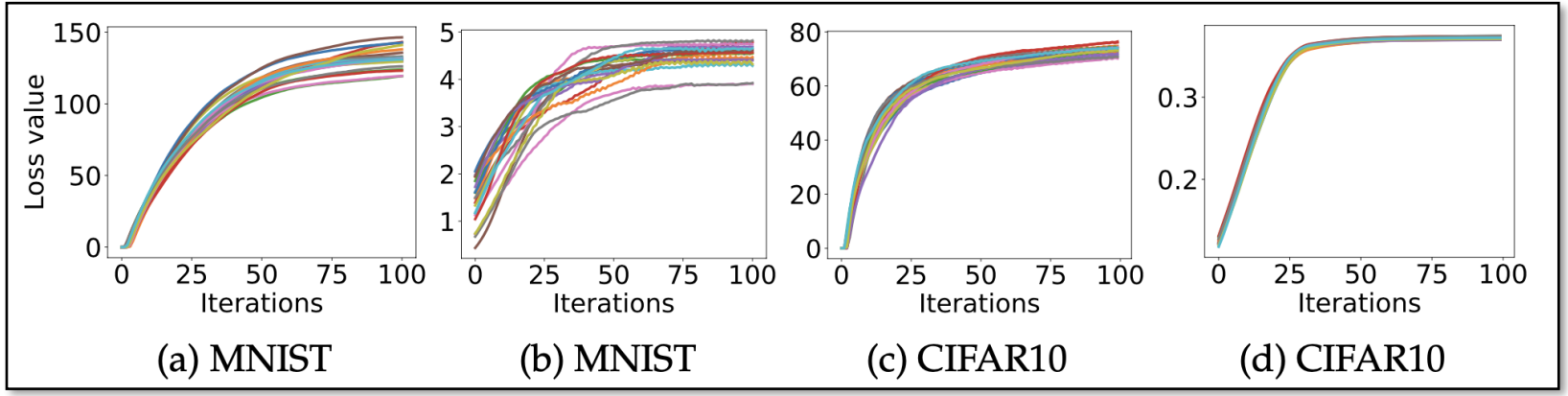
3. Update parameters θ

$$\theta := \theta - \frac{\alpha}{|B|} g$$

Hornik *et al.*, Multilayer feedforward networks are universal approximators, Neural Networks 1989
https://adversarial-ml-tutorial.org/adversarial_training/

MADRY ET AL. – HOW STRONG IS PGD?

- Findings
  - (1, 3) PGD increases the loss values in a fairly consistent way
  - (2, 4) Models trained with PGD attacks are resilient to the same attacks

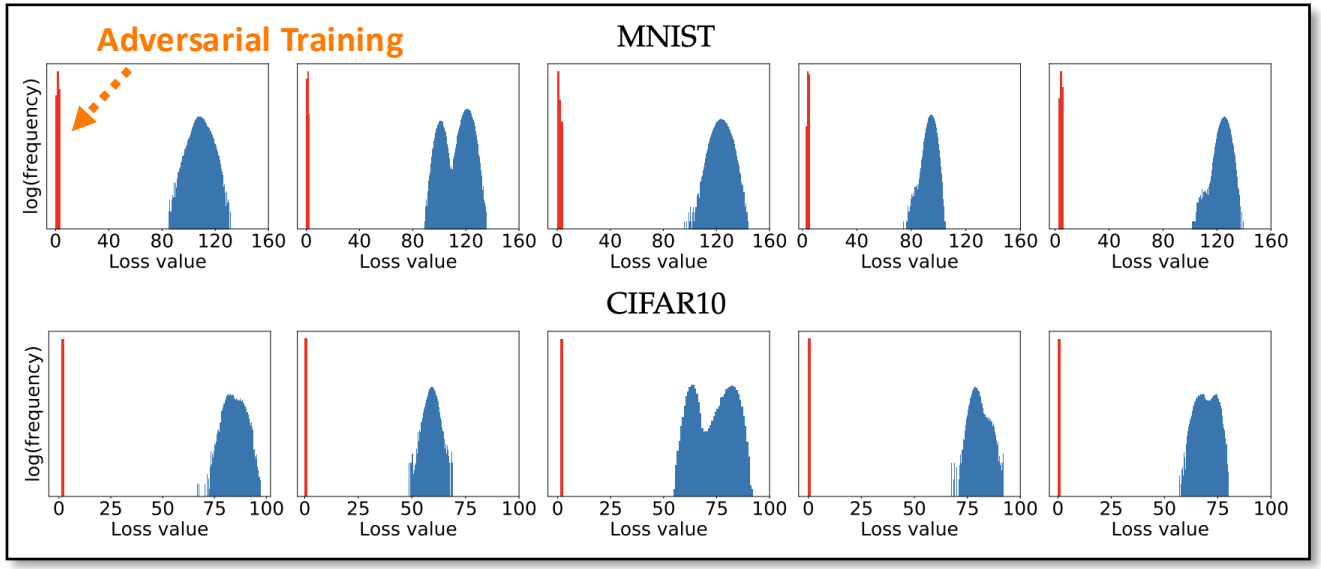


Adversarial Training

Adversarial Training

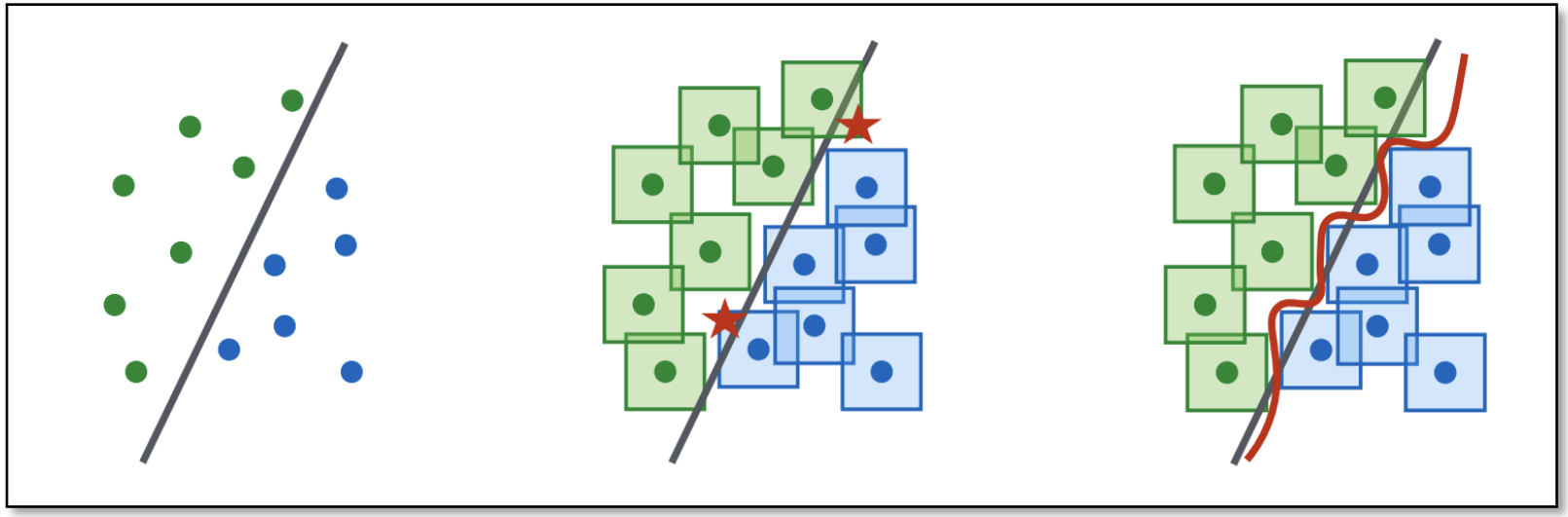
MADRY ET AL. – HOW STRONG IS PGD?

- Findings
  - PGD increases the loss values in a fairly consistent way
  - Models trained with PGD attacks are resilient to the same attacks
  - Final loss of PGD attacks are concentrated (both for defended/undefended models)



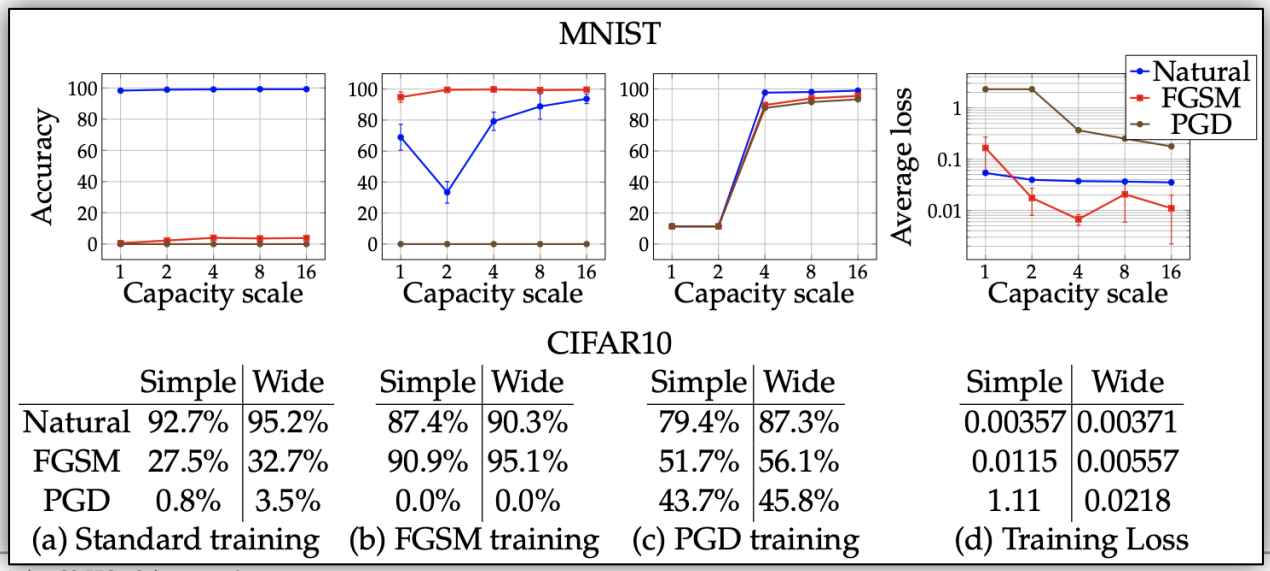
MADRY ET AL. – MODEL CAPACITY

- Why does adversarial training (AT) work?
  - Capacity is crucial for the robustness: robust models need complex decision boundary
  - Capacity alone helps: high-capacity models show more robustness w/o AT



MADRY ET AL. – MODEL CAPACITY

- Why does adversarial training (AT) work? – cont'd
  - Capacity is crucial for the robustness: robust models need complex decision boundary
  - Capacity alone helps: high-capacity models show more robustness w/o AT
  - AT with weak attacks (like FGSM) can't defeat a strong one like PGD
  - (optional) Robustness may be at odds with accuracy



	Simple	Wide	Simple	Wide	Simple	Wide	Simple	Wide
Natural	92.7%	95.2%	87.4%	90.3%	79.4%	87.3%	0.00357	0.00371
FGSM	27.5%	32.7%	90.9%	95.1%	51.7%	56.1%	0.0115	0.00557
PGD	0.8%	3.5%	0.0%	0.0%	43.7%	45.8%	1.11	0.0218
	(a) Standard training		(b) FGSM training		(c) PGD training		(d) Training Loss	

MADRY ET AL. – RESULTS

- Accuracy of the adversarially trained models

Attack	MNIST (ε = 0.3)	CIFAR-10 (ε = 8)
None (clean inputs)	98.8%	87.3%
FGSM	95.6%	56.1%
PGD, white-box (strongest)	89.3%	45.8%
Transfer from another model	95.4%	64.2%

Without adversarial training, MNIST accuracy under FGSM: 6.4%

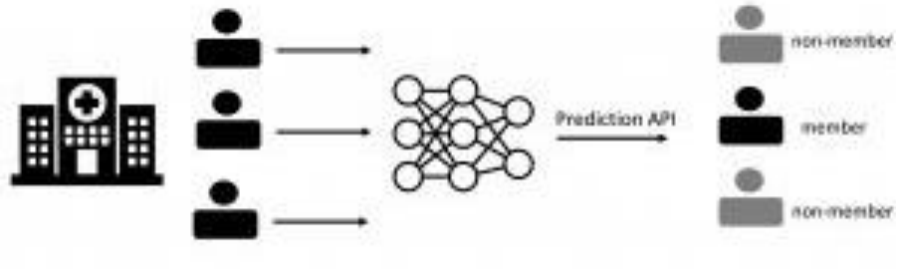
MADRY ET AL. – WHAT REMAINS OPEN

- Lower accuracy on clean inputs
  - CIFAR-10: 95.2% (standard) → 87.3%
- Robust accuracy is still low
  - CIFAR-10: 45.8% under PGD
- Training cost
  - PGD at every training step: about 8× on CIFAR-10
- Only the threat model it trained for
  - Larger ε, ℓ2, rotations, patches are not covered
  - No proof: robust only against attacks we know

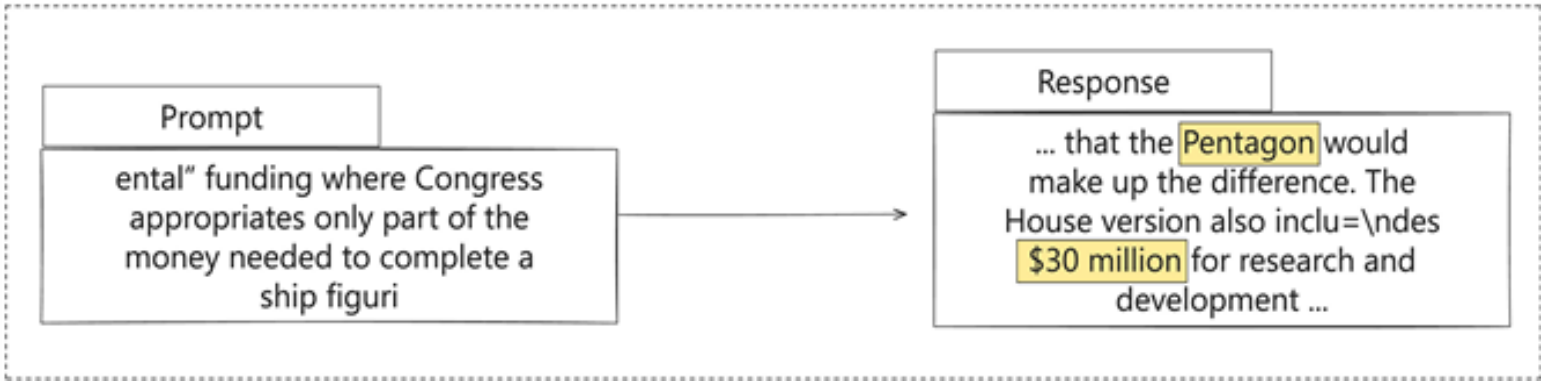
MEMBERSHIP INFERENCE – PRIVACY OF THE TRAINING DATA

PRIVACY IN MACHINE LEARNING

- Membership inference attacks

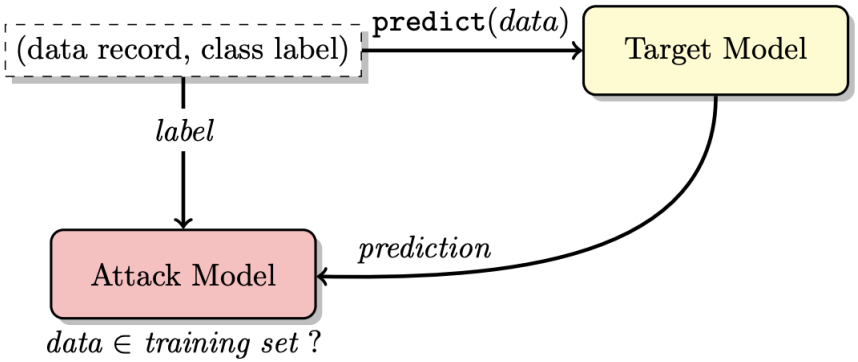


Does the sensitive training set contain a target record?



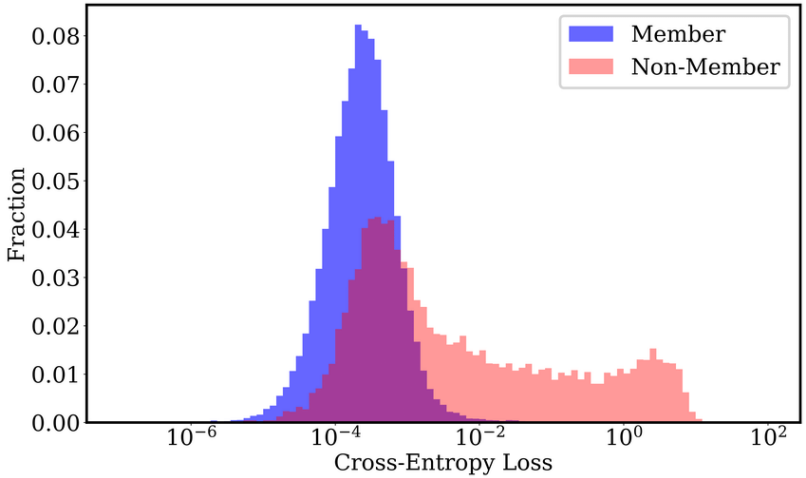
MEMBERSHIP INFERENCE – THREAT MODEL

- Threat model
  - An adversary $\mathcal{A}$ wants to know
  - if a sample $(x, y) \sim D$ is the member of
  - the training set S of an ML model f or not



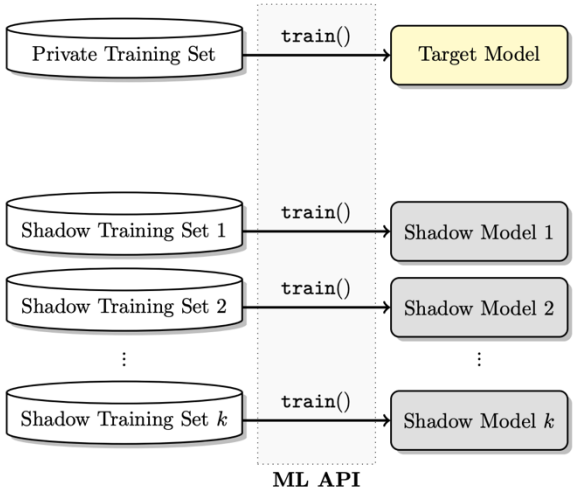
MEMBERSHIP INFERENCE – WHY IT WORKS

- Models fit their training data better
  - Members: low loss
  - Non-members: higher loss
- Simplest attack (Yeom et al.)
  - Query the model on (x, y)
  - Loss below a threshold: guess member



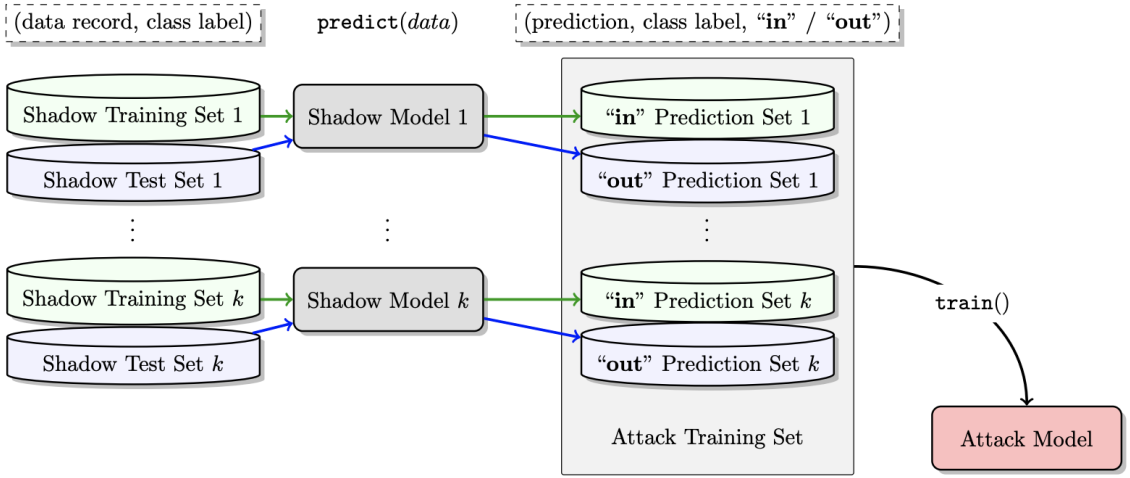
MEMBERSHIP INFERENCE – SHADOW MODELS (SHOKRI ET AL.)

- Shokri *et al.* attack
  - Key idea: shadow models
    - The attacker has some data samples from D
    - If the attacker trains models with those samples, we know their memberships!
    - If shadow models are trained similarity, we can exploit the membership info.!
  - Attacker's data:
    - Know the labeled records: (x, y)
    - Query them to the target model and collect its predictions: $\big((x, y), \hat{y}\big)$
  - How to train?
    - Create a train and test split
    - Use the *train* data to train the shadow models



MEMBERSHIP INFERENCE – SHADOW MODELS (SHOKRI ET AL.)

- Shokri *et al.* attack
  - Attack model
    - Data format $\big((x, y), \hat{y}\big)$
    - Some of them are **"IN"** the shadow train, otherwise **"OUT"**
    - Combine three info. $(y, \hat{y}, \mathbf{IN})$ or $(y, \hat{y}, \mathbf{OUT})$
    - Make the attack model predict **IN** or **OUT**

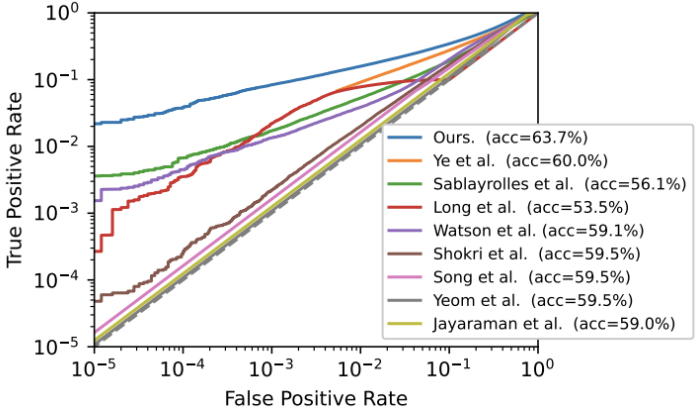


LIRA – RESEARCH QUESTION

- Prior attacks reported about 60% accuracy
  - Looks worrying: better than a coin flip
- Question
  - Can the attack confidently identify any member?
- Approach
  - Measure the attack where it matters: few false positives
  - Judge each example on its own

LIRA – HOW ATTACKS WERE MEASURED

- Metrics for measuring the attack success
  - Membership advantage (Yeom et al.)
  - Precision (Shokri et al.)
  - AUROC (Jayaraman et al.)
  - …

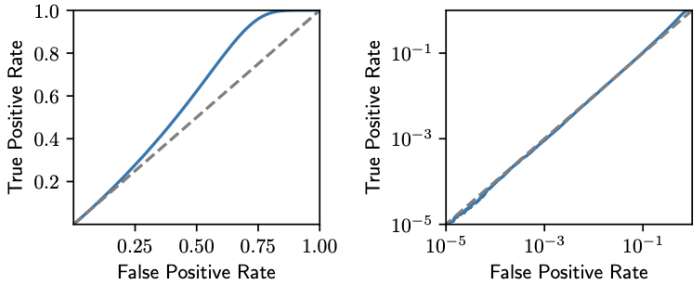


LIRA – WHY AVERAGE ACCURACY MISLEADS

- Example from the paper
  - Attack A: always right on 0.1% of people, a coin flip on the rest
  - Attack B: right 50.05% of the time on everyone
  - Same accuracy; only A breaks anyone's privacy
- LOSS attack on CIFAR-10
  - Accuracy: 60%
  - Its most confident 1% of "member" guesses: right 48%
- Measure TPR at a low FPR (e.g., 0.1%)
  - Plot the ROC curve on a log scale

LIRA – WHY AVERAGE ACCURACY MISLEADS

- Metrics for measuring the attack success
  - Problem of existing metrics
    - Symmetric: equal cost to false-positives and false-negatives
    - Average-case metric: often in security, we are interested in a certain subset
  - LOSS attack
    - Metrics: membership advantage or precision
    - Problem: perform at random at low-FPR



(a) linear scale (b) log scale

Fig. 2: ROC curve for the LOSS baseline membership inference attack, shown with both linear scaling (left), also and log-log scaling (right) to emphasize the low-FPR regime.

LIRA – PER-EXAMPLE HARDNESS

- LiRA (The likelihood ratio attack)
  - Per-sample hardness score
    - Not all examples are equal
    - Some samples are easier to fit
    - Some samples have a larger separability
    - It does not matter if it is an inlier or outlier

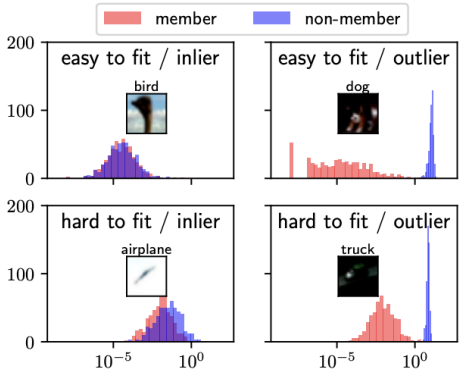
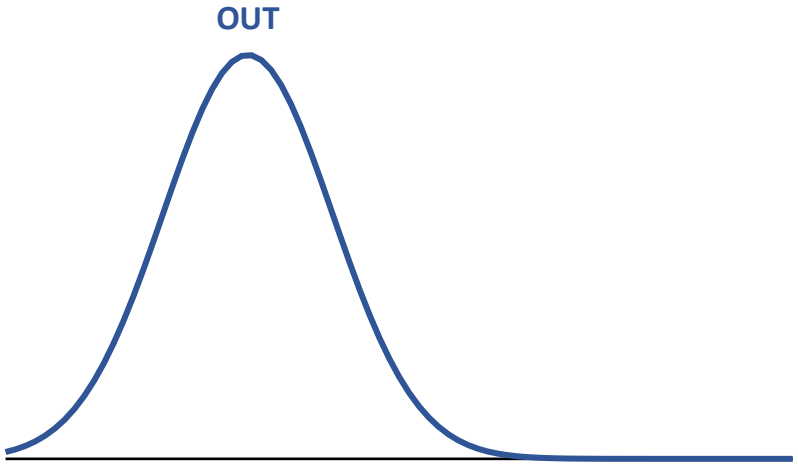


Fig. 3: Some examples are easier to fit than others, and some have a larger separability between their losses when being a member of the training set or not. We train 1024 models on random subsets of CIFAR-10 and plot the losses for four examples when the example is a member of the training set ($\tilde{\mathbb{Q}}_{\text{in}}(x, y)$, in red) or not ($\tilde{\mathbb{Q}}_{\text{out}}(x, y)$, in blue).

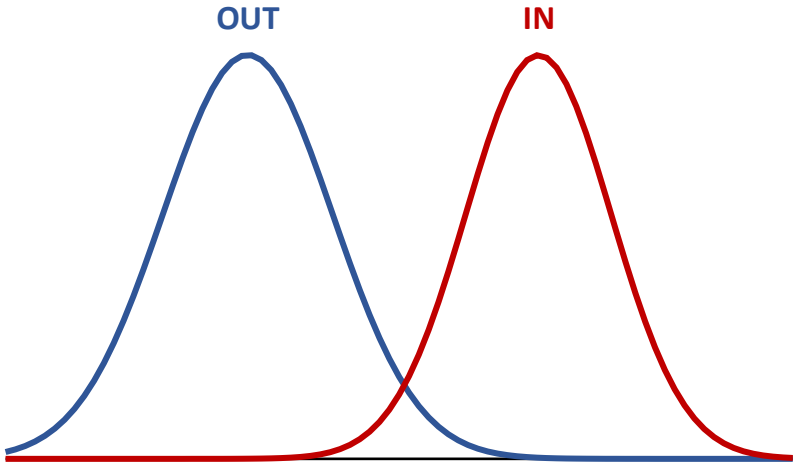
LIRA – THE LIKELIHOOD RATIO ATTACK

- Is (x, y) in the training set?
  - Train shadow models without (x, y)
  - Query each on x: its confidence
  - Fit a Gaussian: OUT



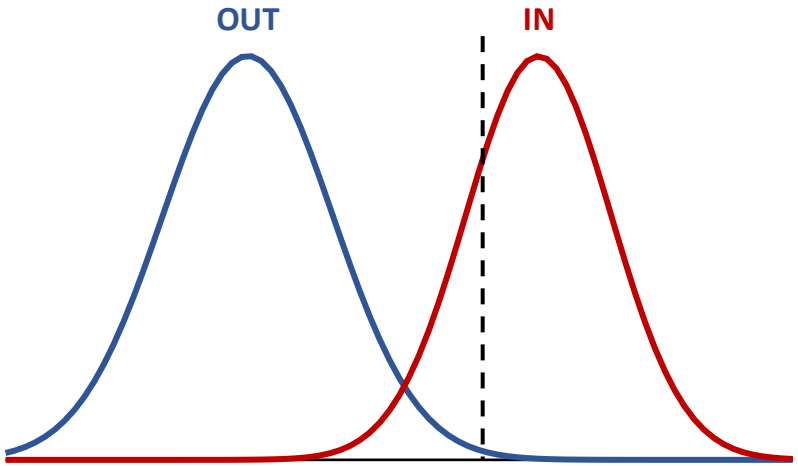
LIRA – THE LIKELIHOOD RATIO ATTACK – CONT'D

- Is (x, y) in the training set?
  - Train shadow models without (x, y)
  - Query each on x: its confidence
  - Fit a Gaussian: OUT
  - Train shadow models with (x, y)
  - Fit a Gaussian: IN



LIRA – THE LIKELIHOOD RATIO ATTACK – CONT'D

- Is (x, y) in the training set?
  - Train shadow models without (x, y)
  - Query each on x: its confidence
  - Fit a Gaussian: OUT
  - Train shadow models with (x, y)
  - Fit a Gaussian: IN
- Query the target model on x
  - Λ = p(conf | IN) / p(conf | OUT)
  - Large Λ: likely a member



LiRA – THE LIKELIHOOD RATIO ATTACK

- LiRA (The likelihood ratio attack)
  - Per-sample hardness score
    - Not all examples are equal
    - Some samples are easier to fit
    - Some samples have a larger separability
    - It does not matter if it is an inlier or outlier
  - Proposed attack
    - Compute per-sample hardness scores
    - Use parametric modeling

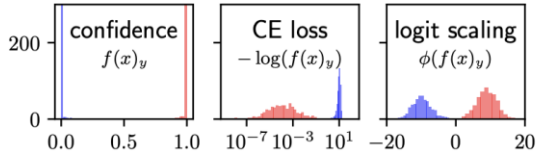


Fig. 4: The model's confidence, or its logarithm (the cross-entropy loss) are not normally distributed. Applying the logit function yields values that are approximately normal.

Algorithm 1 Our online Likelihood Ratio Attack (LiRA). We train shadow models on datasets with and without the target example, estimate mean and variance of the loss distributions, and compute a likelihood ratio test. (In our **offline** variant, we omit lines 5, 6, 10, and 12, and instead return the prediction by estimating a single-tailed distribution, as is shown in Equation (4).)

Require: model f, example (x, y), data distribution $\mathbb{D}$

1: $\text{confs}_{\text{in}} = \{\}$
2: $\text{confs}_{\text{out}} = \{\}$
3: **for** N times **do**
4: $\quad D_{\text{attack}} \leftarrow^{\$} \mathbb{D}$ ▷ *Sample a shadow dataset*
5: $\quad f_{\text{in}} \leftarrow \mathcal{T}(D_{\text{attack}} \cup \{(x, y)\})$ ▷ *train IN model*
6: $\quad \text{confs}_{\text{in}} \leftarrow \text{confs}_{\text{in}} \cup \{\phi(f_{\text{in}}(x)_y)\}$
7: $\quad f_{\text{out}} \leftarrow \mathcal{T}(D_{\text{attack}} \backslash \{(x, y)\})$ ▷ *train OUT model*
8: $\quad \text{confs}_{\text{out}} \leftarrow \text{confs}_{\text{out}} \cup \{\phi(f_{\text{out}}(x)_y)\}$
9: **end for**
10: $\mu_{\text{in}} \leftarrow \texttt{mean}(\text{confs}_{\text{in}})$
11: $\mu_{\text{out}} \leftarrow \texttt{mean}(\text{confs}_{\text{out}})$
12: $\sigma^2_{\text{in}} \leftarrow \texttt{var}(\text{confs}_{\text{in}})$
13: $\sigma^2_{\text{out}} \leftarrow \texttt{var}(\text{confs}_{\text{out}})$
14: $\text{conf}_{\text{obs}} = \phi(f(x)_y)$ ▷ *query target model*
15: **return** $\Lambda = \dfrac{p(\text{conf}_{\text{obs}} \mid \mathcal{N}(\mu_{\text{in}}, \sigma^2_{\text{in}}))}{p(\text{conf}_{\text{obs}} \mid \mathcal{N}(\mu_{\text{out}}, \sigma^2_{\text{out}}))}$

LIRA – EVALUATION SETUP

- Setup
  - Datasets: CIFAR-10, CIFAR-100, ImageNet and WikiText
  - Models
    - Wide-ResNet (CIFAR-10 and -100)
    - ResNet-50 (ImageNet)
    - GPT-2 small (WikiText)
  - LiRA setup
    - Shadow models: 64 for ImageNet and 256 for others
    - Target model: trained on half of each dataset
  - Metric
    - TPR at 0.1% and 0.001% FPR
    - ROC curve

LIRA – RESULTS

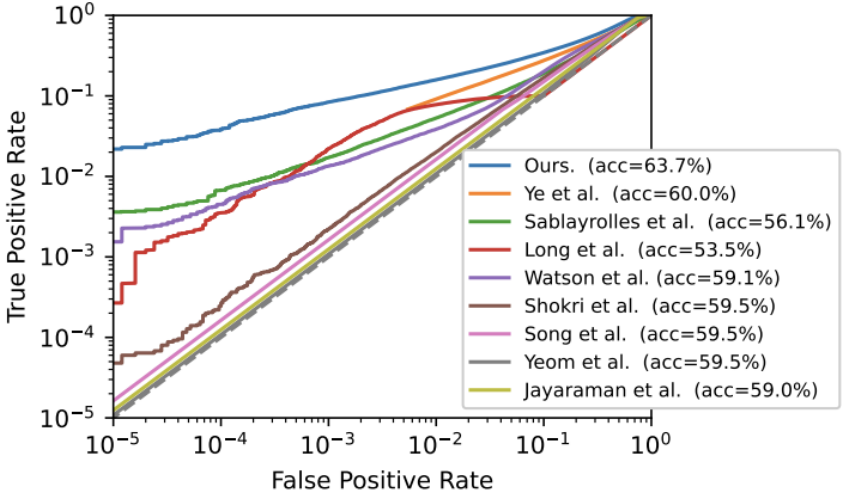
- LiRA (online) attack vs others

Method	shadow models	multiple queries	class hardness	example hardness	TPR @ 0.001% FPR			TPR @ 0.1% FPR			Balanced Accuracy		
					C-10	C-100	WT103	C-10	C-100	WT103	C-10	C-100	WT103
Yeom et al. [70]	○	○	○	○	0.0%	0.0%	0.00%	0.0%	0.0%	0.1%	59.4%	78.0%	50.0%
Shokri et al. [60]	●	○	●	○	0.0%	0.0%	–	0.3%	1.6%	–	59.6%	74.5%	–
Jayaraman et al. [25]	○	●	○	○	0.0%	0.0%	–	0.0%	0.0%	–	59.4%	76.9%	–
Song and Mittal [61]	●	○	●	○	0.0%	0.0%	–	0.1%	1.4%	–	59.5%	77.3%	–
Sablayrolles et al. [56]	●	○	●	●	0.1%	0.8%	0.01%	1.7%	7.4%	1.0%	56.3%	69.1%	**65.7%**
Long et al. [37]	●	○	●	●	0.0%	0.0%	–	2.2%	4.7%	–	53.5%	54.5%	–
Watson et al. [68]	●	○	●	●	0.1%	0.9%	0.02%	1.3%	5.4%	1.1%	59.1%	70.1%	65.4%
Ye et al. [69]	●	○	●	●	-	-	-	-	-	-	60.3%	76.9%	65.5%
Ours	●	●	●	●	**2.2%**	**11.2%**	**0.09%**	**8.4%**	**27.6%**	**1.4%**	**63.8%**	**82.6%**	65.6%

TABLE I: **Comparison of prior membership inference attacks** under the same settings for well-generalizing models on CIFAR-10, CIFAR-100, and WikiText-103 using 256 shadow models. Accuracy is only presented for completeness; we do not believe this is a meaningful metric for evaluating membership inference attacks. Full ROC curves are presented in Appendix A.

LIRA – RESULTS

- LiRA (online) attack vs others
  - 10× more successful than prior attacks at low FPRs



LIRA – DOES GENERALIZATION PROTECT?

- LiRA (online) attack and the generalization gap
  - Overfit models tend to be more vulnerable
  - Same gap, up to 100× different TPR
  - Better test accuracy does not mean safer

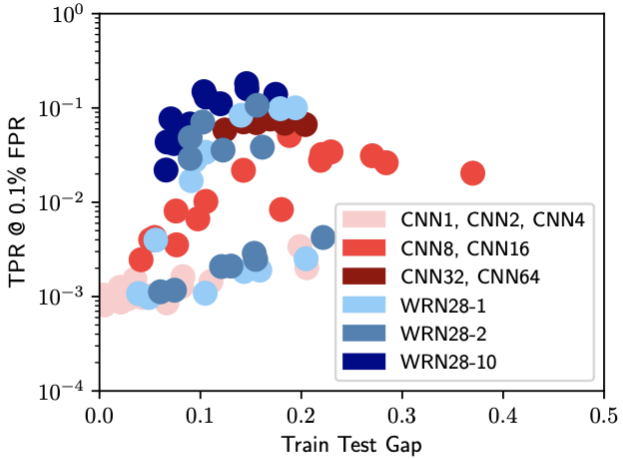
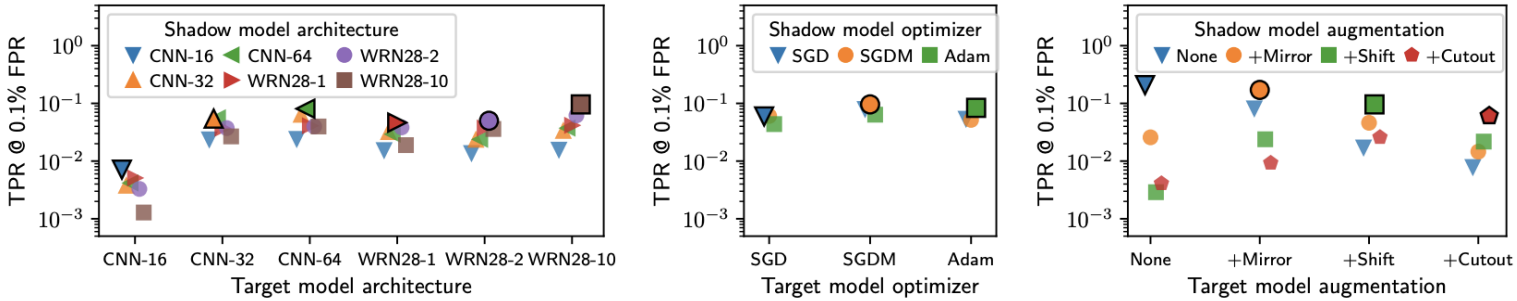


Fig. 7: Attack true-positive rate versus model train-test gap for a variety of CIFAR-10 models.

LIRA – WHAT IF THE ATTACKER GUESSES WRONG?

- LiRA (online) attack with different settings
  - While the training configurations are different from shadow models to the target
  - LiRA attack performs consistently; the attack is agnostic to the training setups



(a) Vary model architecture. (b) Vary training optimizer. (c) Vary data augmentation.

Fig. 11: Our attack succeeds when the adversary is uncertain of the target model's training setup. We vary the target model's architecture (a), the training optimizer (b) and the data augmentation (c), as well as the adversary's guess of each of these properties when training shadow models. The attack performs best when the adversary guesses correctly (black-lined markers).

LIRA – THREAT MODEL

- Goal
  - Tell whether one record (x, y) was in the training set
- Knowledge
  - Data from the same distribution
  - Roughly how the model was trained
- Capabilities
  - Query the model and read its confidence scores
  - Train many shadow models (64 to 256)

LIRA – COUNTERMEASURES

- What can we do?

Attacker needs	Countermeasure	Problem
Confidence scores	Return only the label	Label-only attacks
A gap on each example	Regularization, early stopping	Outliers still leak
Effect of one record	Differential privacy (DP-SGD)	Lower accuracy
Many shadow models	Keep the training data secret	Public data is enough

RECAP – WHERE WE ARE

- Part I – Network/Internet security
  - The attacker is outside your machine
- Part II – Computer systems security
  - The attacker is inside your machine
  - Memory safety: how the attacker gets in
  - OS security: decide what the attacker may do (access control)
- Part III – Isolation and (software-induced) breaks
  - The attacker is inside your program or is the OS itself
  - The attacker exploits your hardware and breaks integrity and confidentiality
- Part IV – Software/Web security
  - The attacker turns bugs into exploits and delivers them through the web
- Part V – Trustworthy ML
  - The attacker targets ML models: their inputs and their training data

Thank You!

Sanghyun Hong

https://true-lab.ai/courses/Sec-Grad/current



