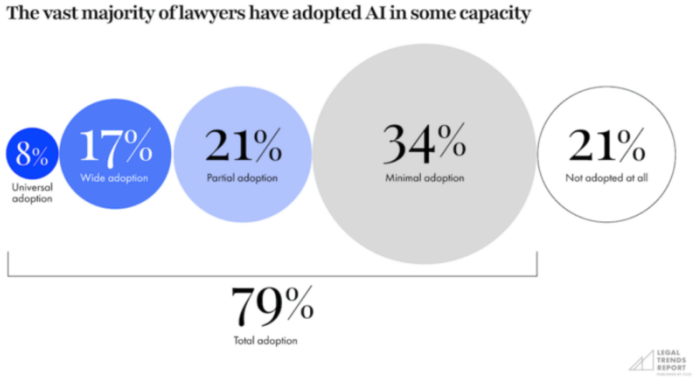
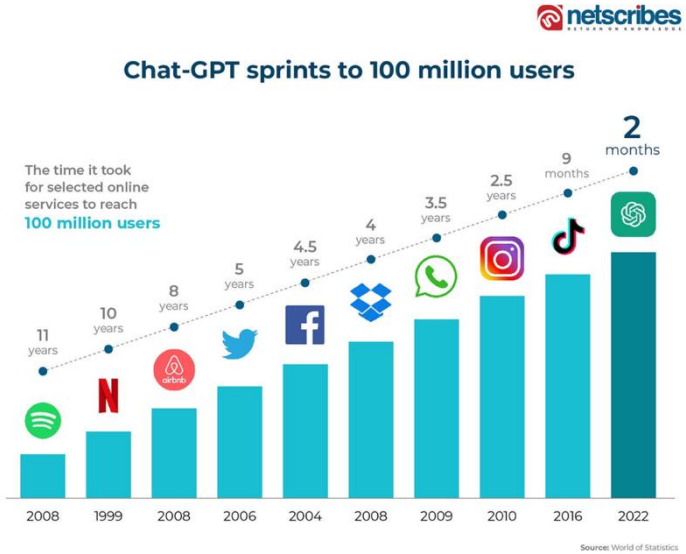
AI 539: TRUSTWORTHY ML
PART IV – LANGUAGE MODEL PRIVACY

Sanghyun Hong
sanghyun.hong@oregonstate.edu

MACHINE LEARNING WILL BE EVERYWHERE

- Market size is expected to reach USD $33.8B by 2030[1]

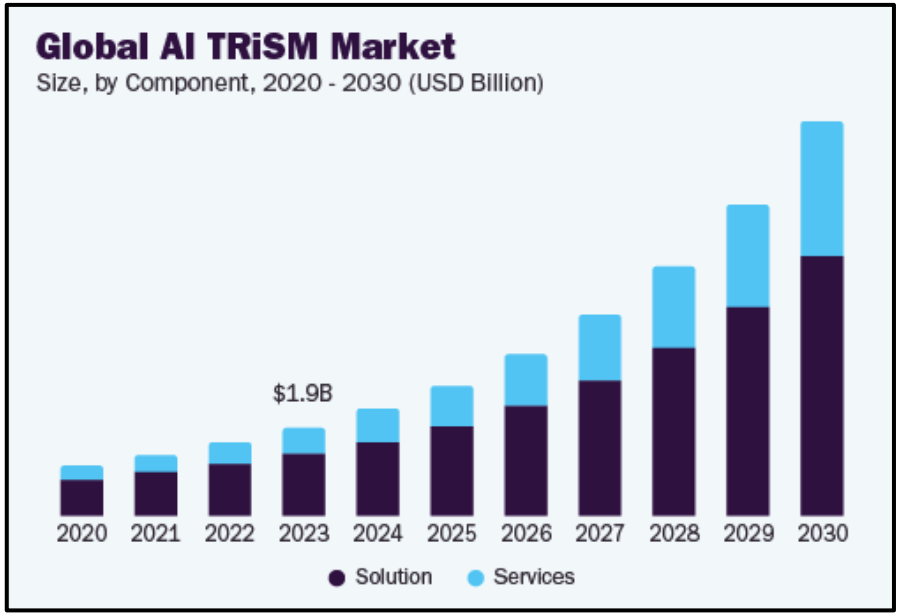


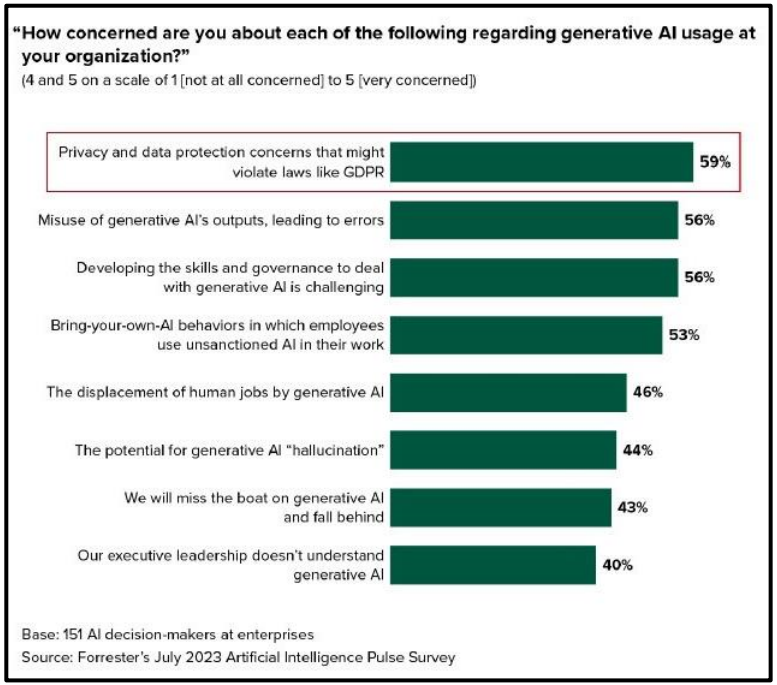


[1]https://www.grandviewresearch.com/industry-analysis/us-generative-ai-market-report
[2]https://www.clio.com/blog/highlights-from-2024-legal-trends-report/
[3]https://ai.plainenglish.io/chat-gpt-achieving-100-million-users-in-just-2-month-a-deep-analysis-a453e6f85acf?gi=4fa59a4a8c9d

PRIVACY IS ONE OF THE GROWING CONCERNS IN ML ECOSYSTEM

- AI TRISM market size is expected to reach USD $5.4B by 2030[1]
- GDPR, CCPA, CRPA, HIPAA, or FTC





[2]https://www.grandviewresearch.com/industry-analysis/ai-trust-risk-security-management-market-report

How to measure privacy of language models?

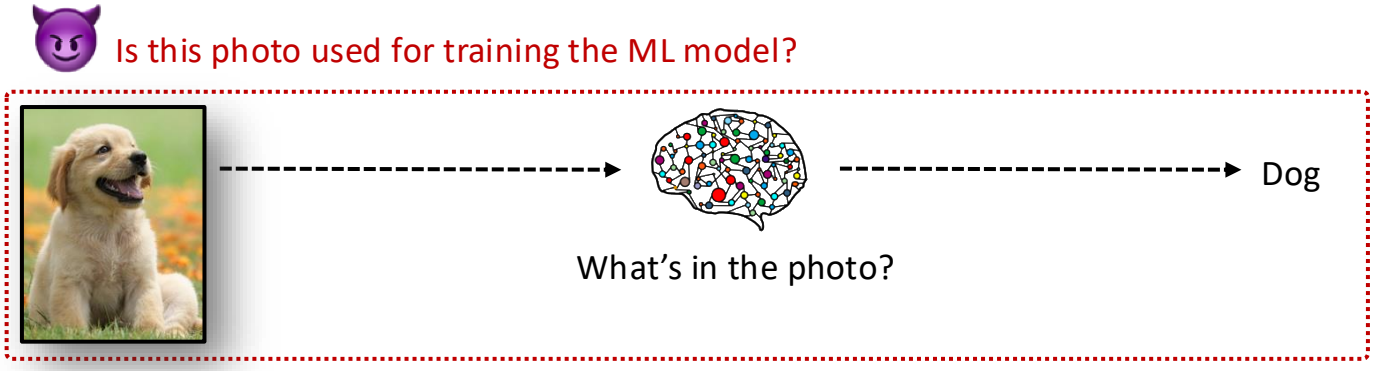
- Measure memorization[1]
  - Memorization is *not* inherently problematic
  - Privacy risk arises when it influences model outputs

$$\texttt{mem}(\mathcal{A}, S, i) := \Pr_{h \leftarrow \mathcal{A}(S)}[h(x_i) = y_i] - \Pr_{h \leftarrow \mathcal{A}(S^{\setminus i})}[h(x_i) = y_i]$$

[1]Feldman and Zhang, When Neural Networks Memorize and Why? Discovering the Long Tail via Influence Estimation, NeurIPS 2020

How to measure the memorization?

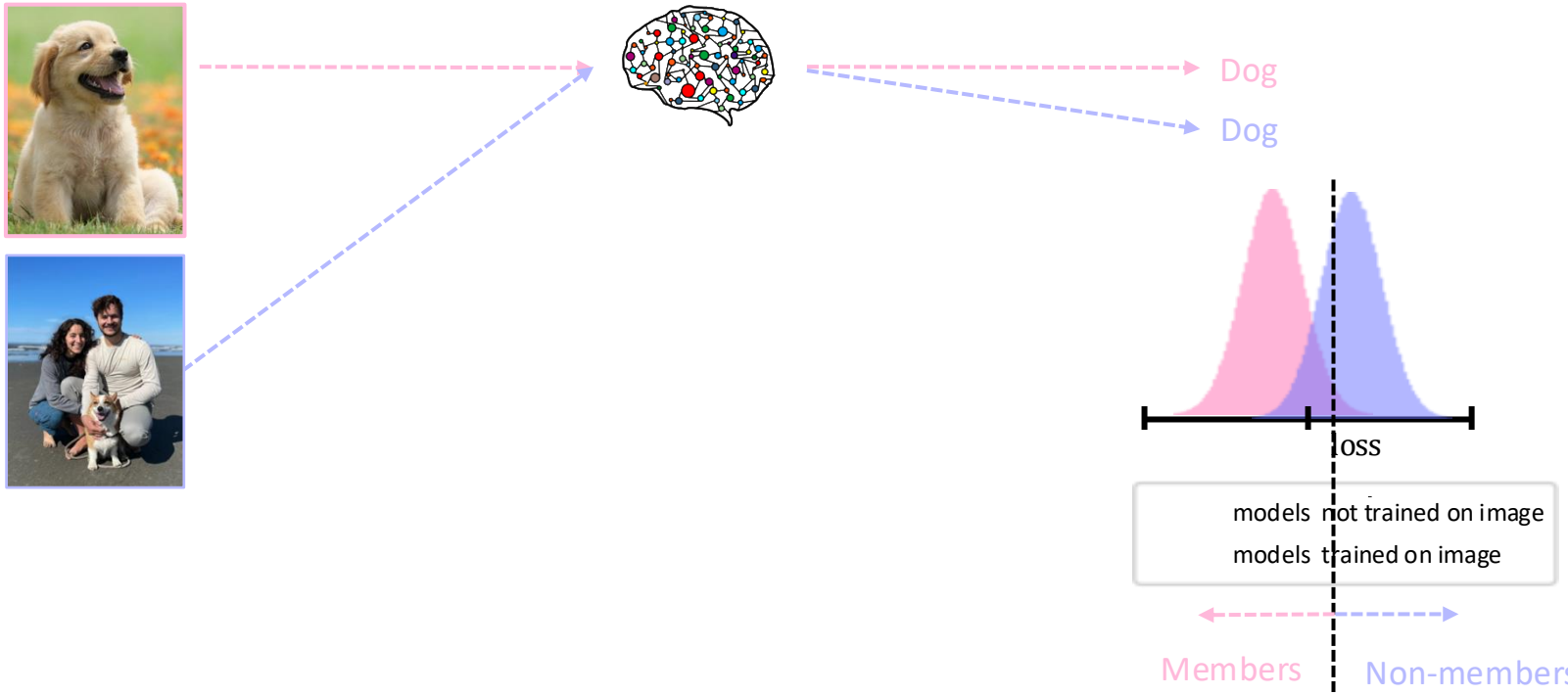
- Prior work proposes privacy attacks
  - Membership (or attribute) inference attacks[1]
  - Model inversion[2] and data extraction attacks[3,4]



[1]Carlini et al., Membership Inference Attacks From First Principles
[2]Fredrikson et al., Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures
[3]Keum et al, Private Investigator: Extracting Personally Identifiable Information from LLMs Using Optimized Prompts
[4]Nasr et al., Scalable Extraction of Training Data from Aligned, Production Language Models

How does an adversary identify the membership?

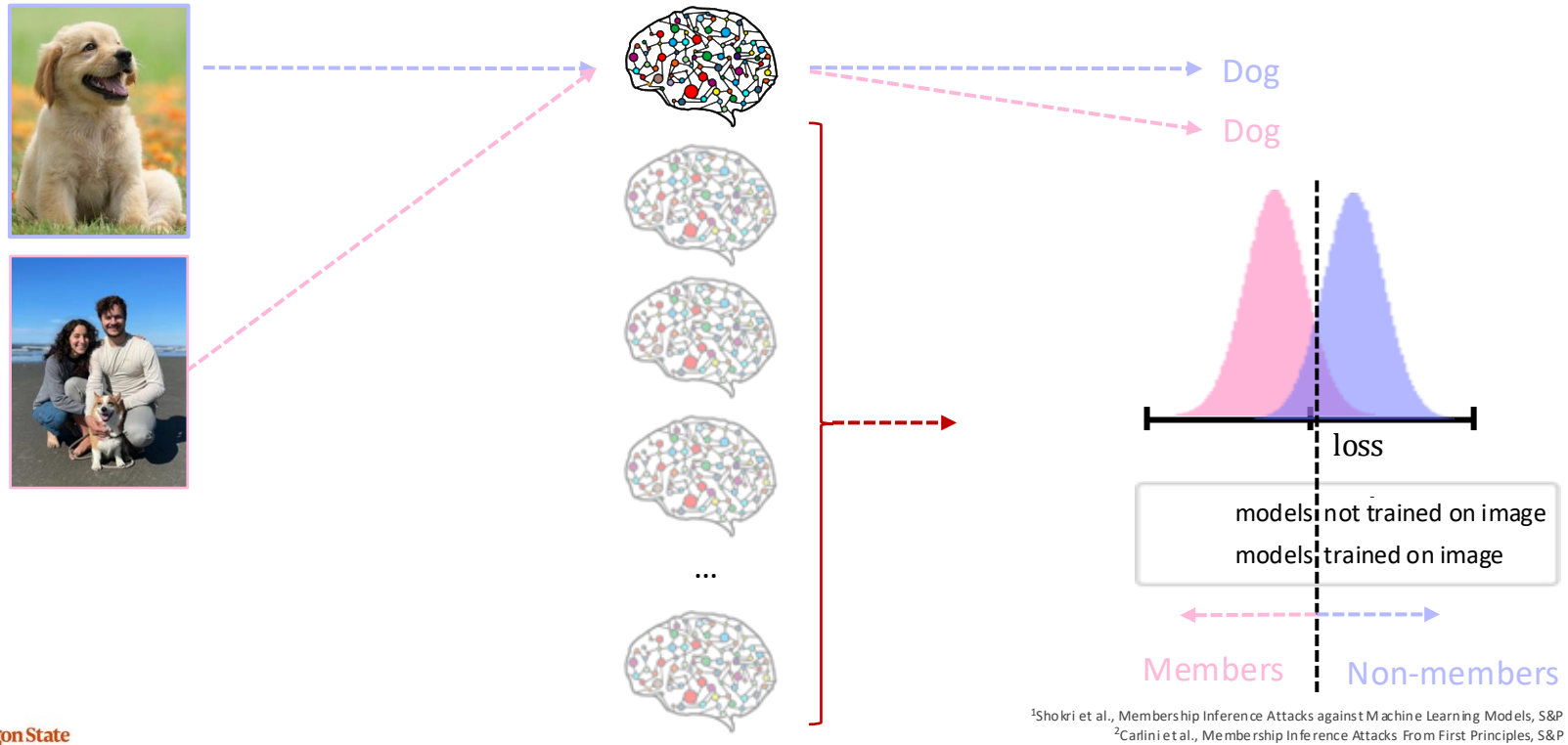
- Initial attacks[1] use the confidence (loss)-based thresholding



[1]Yeom et al., Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting

How does an adversary identify the membership?

- Subsequent attacks[1] use the shadow models for calibrating this threshold

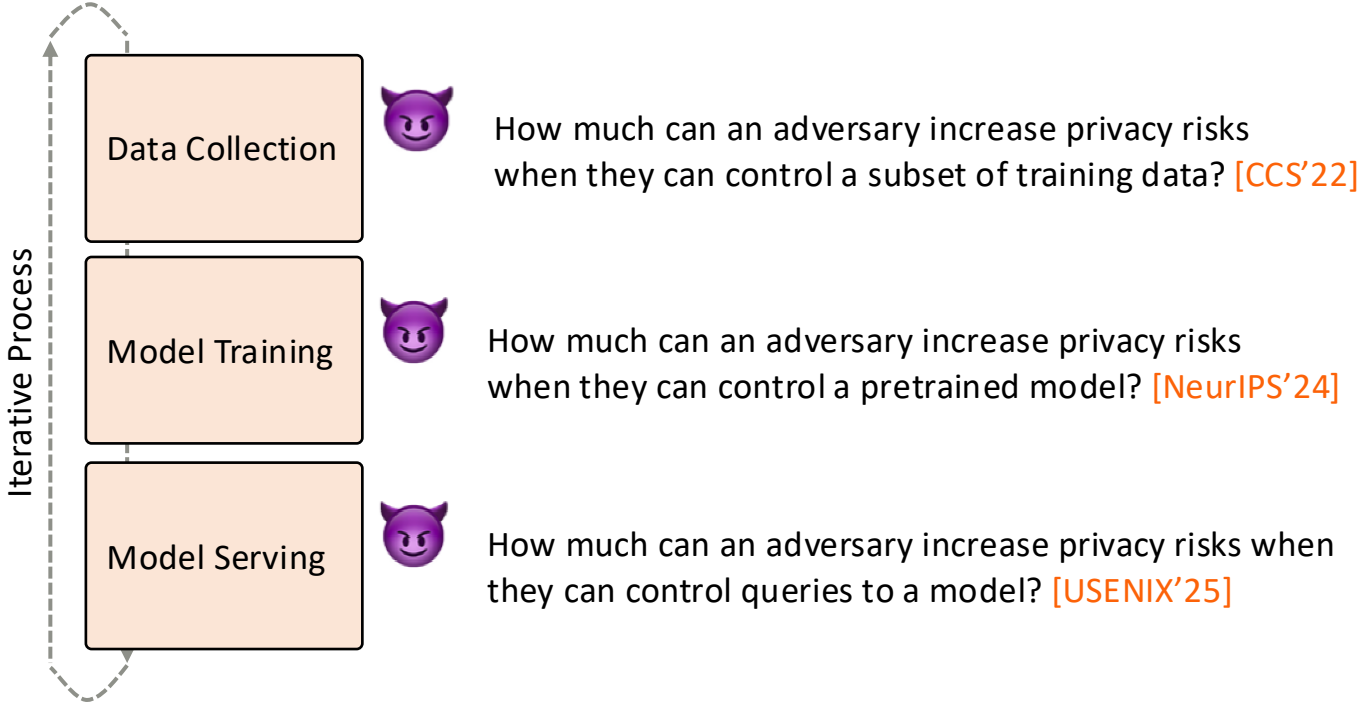


[1]Shokri et al., Membership Inference Attacks against Machine Learning Models, S&P 2017
[2]Carlini et al., Membership Inference Attacks From First Principles, S&P 2022

PRIVACY IS *A SYSTEMS SECURITY PROBLEM*, NOT SOLELY AN ML PROBLEM

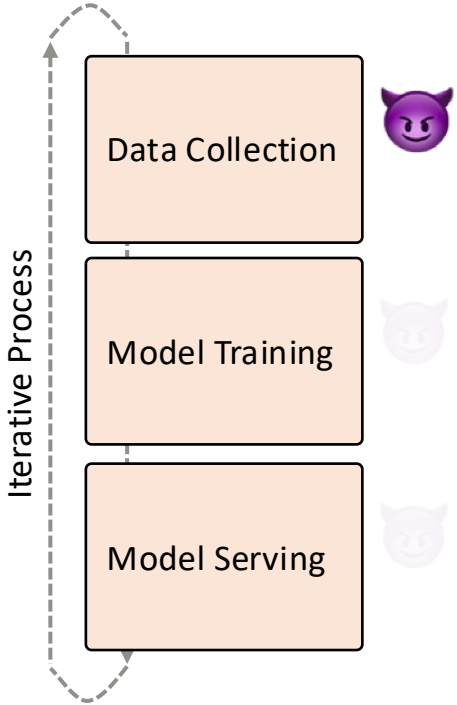
PRIVACY IS A SYSTEMS SECURITY PROPERTY

- Privacy risks emerge from the pipeline



PRIVACY IS A SYSTEMS SECURITY PROPERTY

- Privacy risks emerge from the pipeline



How much can an adversary increase privacy risks when they can control a subset of training data? [CCS'22]

How much can an adversary increase privacy risks when they can control a pretrained model? [NeurIPS'24]

How much can an adversary increase privacy risks when they can control queries to a model? [USENIX'25]

ML MODELS LEARN FROM "DATA"

Computer security seeks to ensure a system's *integrity* against attackers by creating clear boundaries between the system and the outside world. In ML, however, the most critical ingredient of all–the training data–comes directly from the outside world.

– Bishop (2002)[1]

[1]Bishop et al, The Art and Science of Computer Security, 2002

DATA POISONING ATTACKS IN ML

Computer security seeks to ensure a system's *integrity* against attackers by creating clear boundaries between the system and the outside world. In ML, however, the most critical ingredient of all–the training data–comes directly from the outside world. … an attacker can inject malicious data … such data poisoning requires us to re-think what it means for a system to be secure.

– Steinhardt, Koh, and Liang (2017)[1]

[1]Steinhardt et al., Certified Defenses for Data Poisoning Attacks, NeurIPS 2017

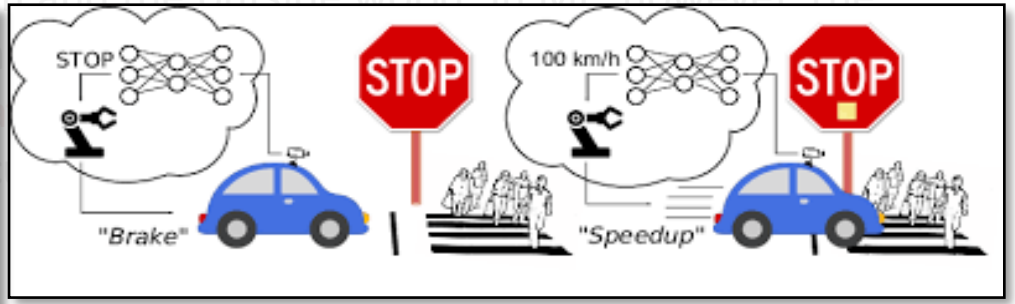
DATA POISONING ATTACKS IN ML

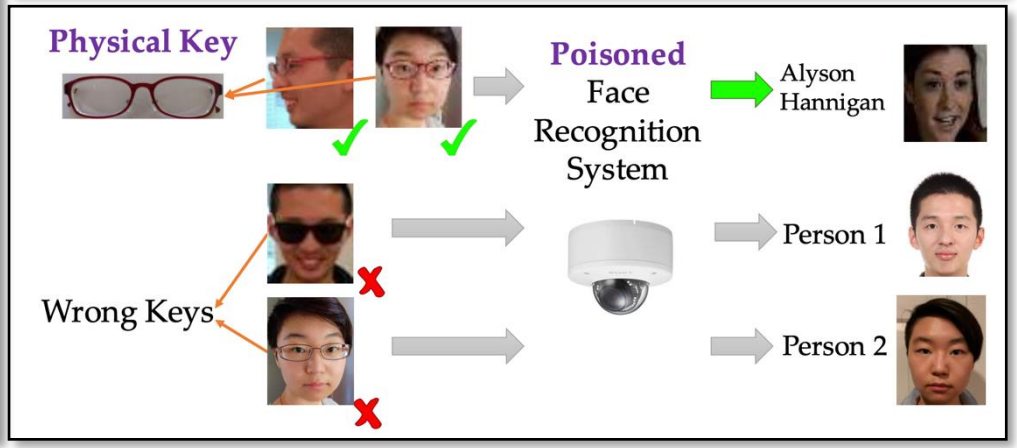
Computer security seeks to ensure a system's *integrity* against attackers by creating clear boundaries between the system and the outside world. In ML, however, the

MICROSOFT / WEB / TL;DR

Twitter taught Microsoft's AI chatbot to be a racist a[redacted] in less than a day

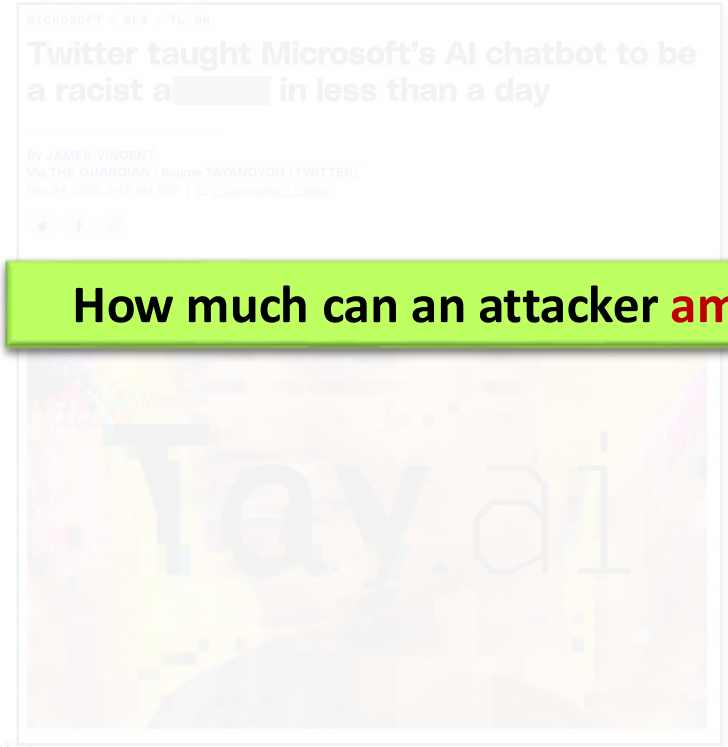
By JAMES VINCENT
Via THE GUARDIAN | Source TAYANDYOU (TWITTER)
Mar 24, 2016, 3:43 AM PDT | 0 Comments / 0 New

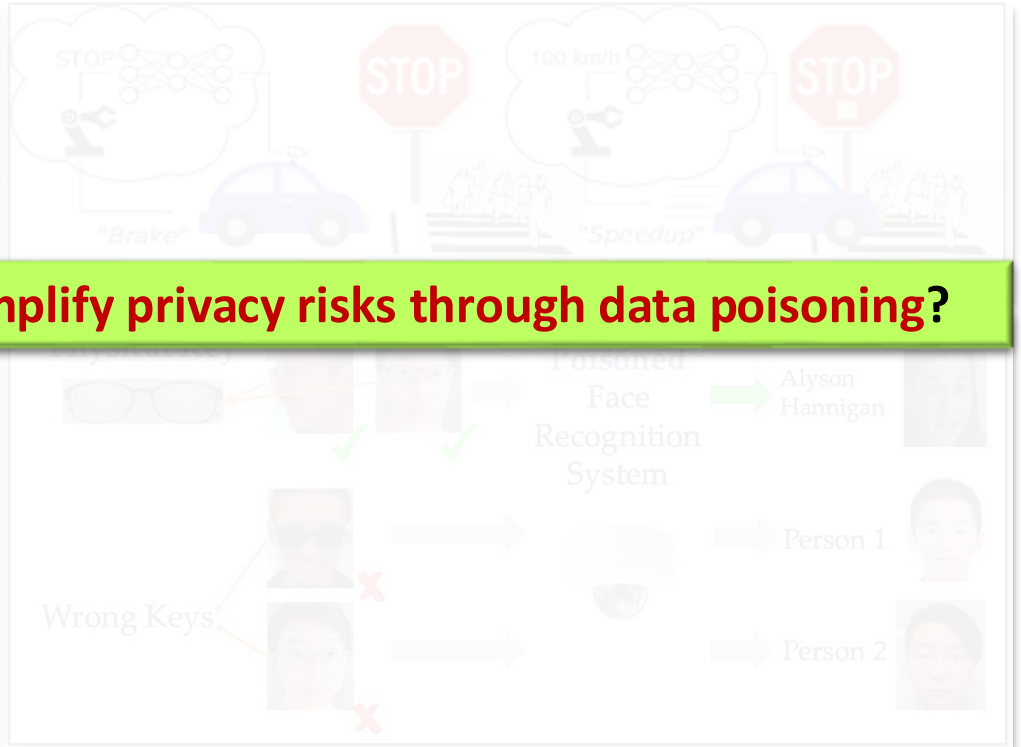




DATA POISONING ATTACKS IN ML – CONT'D

Computer security seeks to ensure a system's *integrity* against attackers by creating

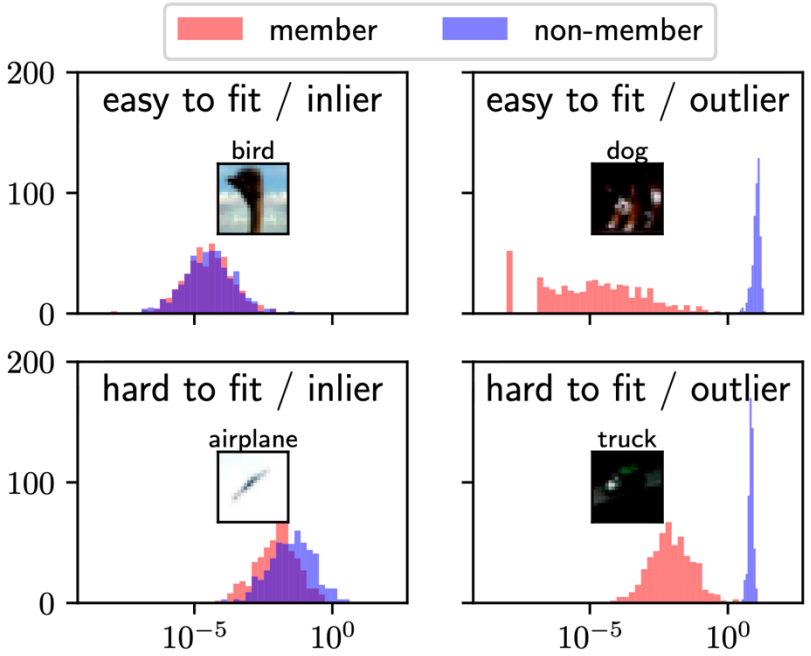




How much can an attacker amplify privacy risks through data poisoning?

When does membership inference work?

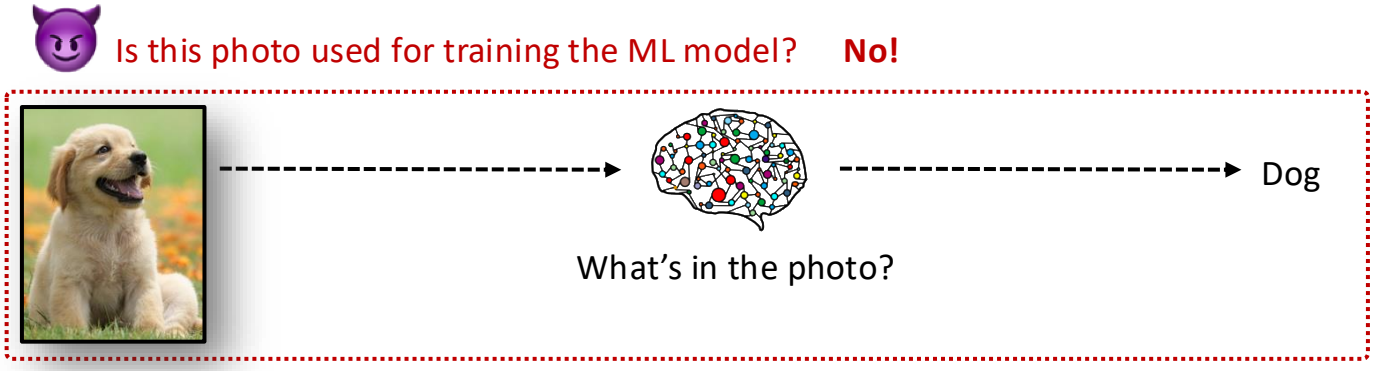
- Prior work[3] showed that
  - This is *not* about the difficulty in learning
  - This is about the *outliers* (defined by the loss when not trained on that sample)



[3] Carlini et al., Membership Inference Attacks from First Principles

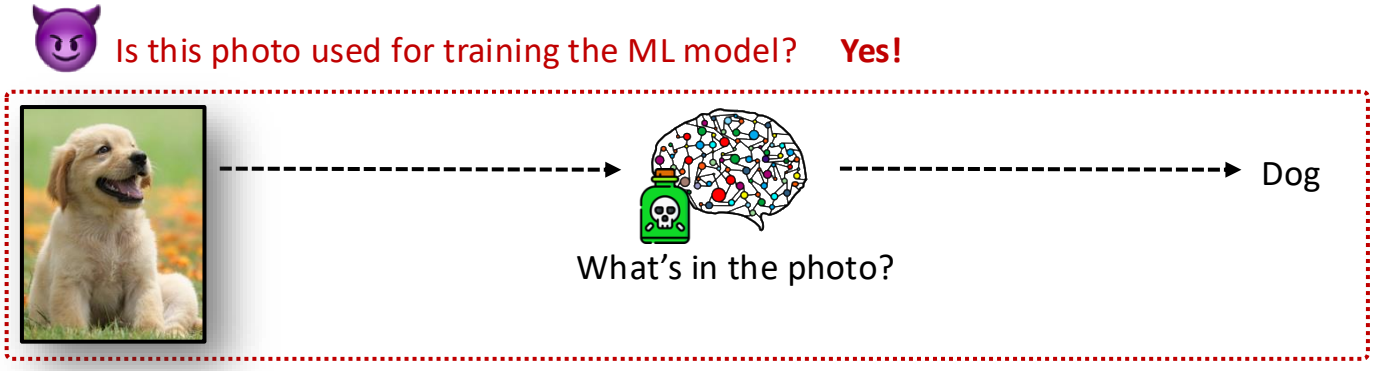
A NEW THREAT: PRIVACY POISONING

- Can we make inliers to outliers through data poisoning?



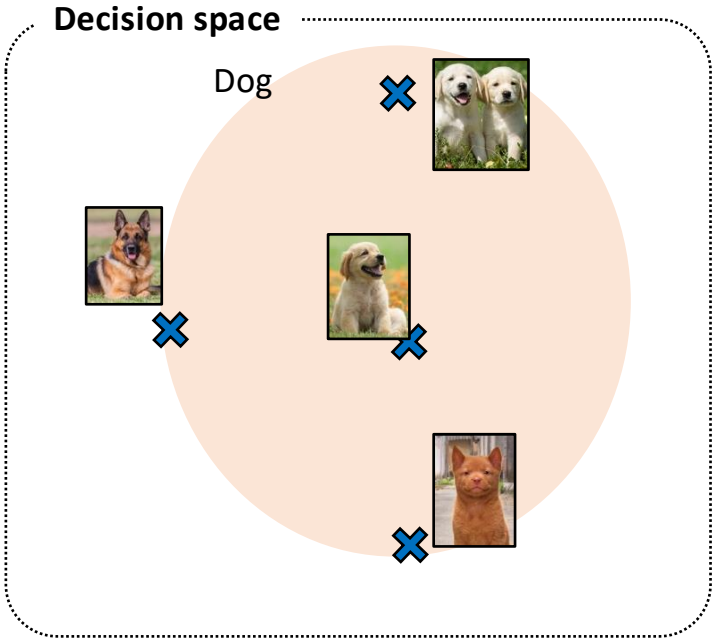
A NEW THREAT: PRIVACY POISONING

- Can we make inliers to outliers through data poisoning?



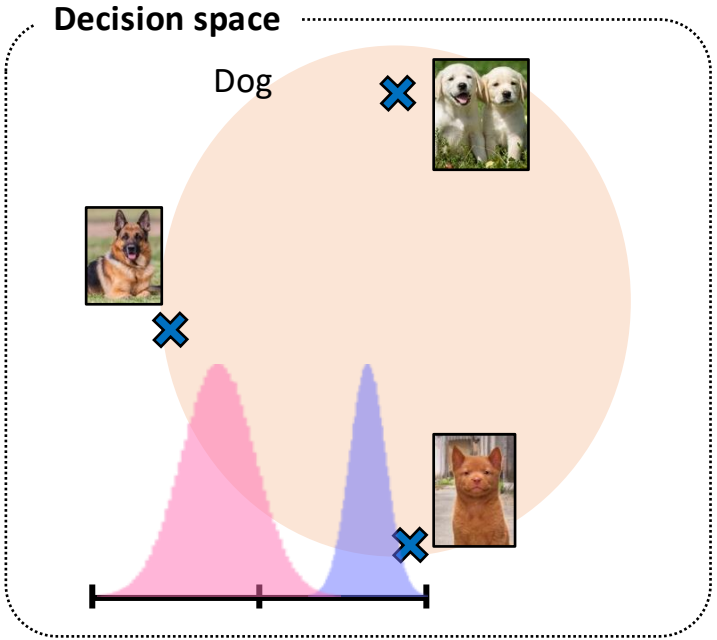
How does our privacy poisoning work?

- Turn inliers to outliers



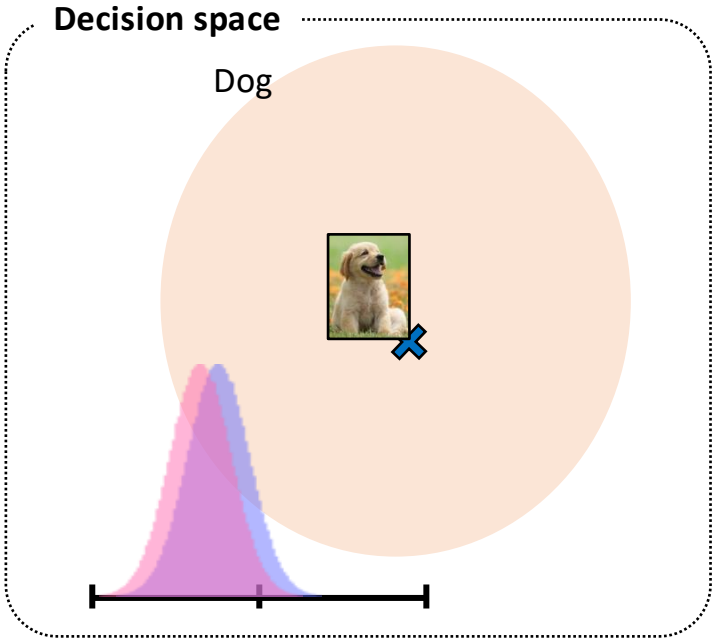
How does our privacy poisoning work?

- Turn inliers to outliers



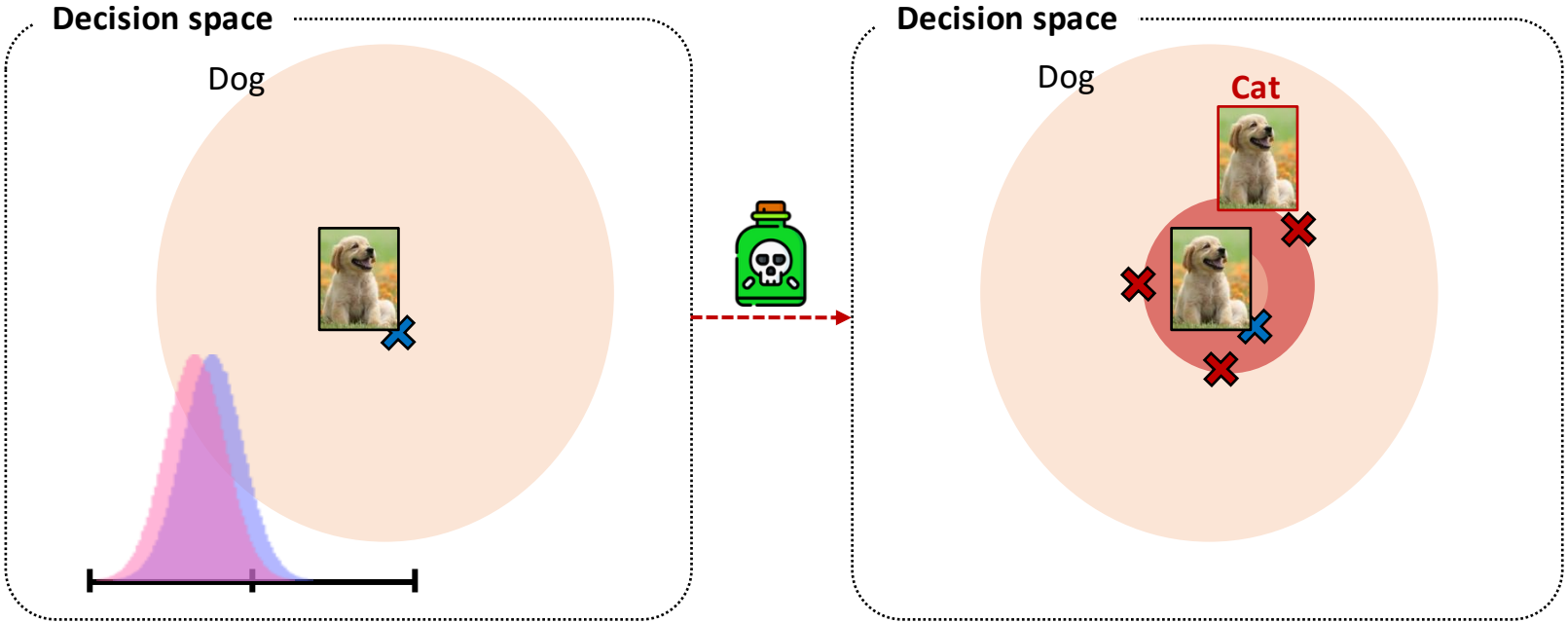
How does our privacy poisoning work?

- Turn inliers to outliers



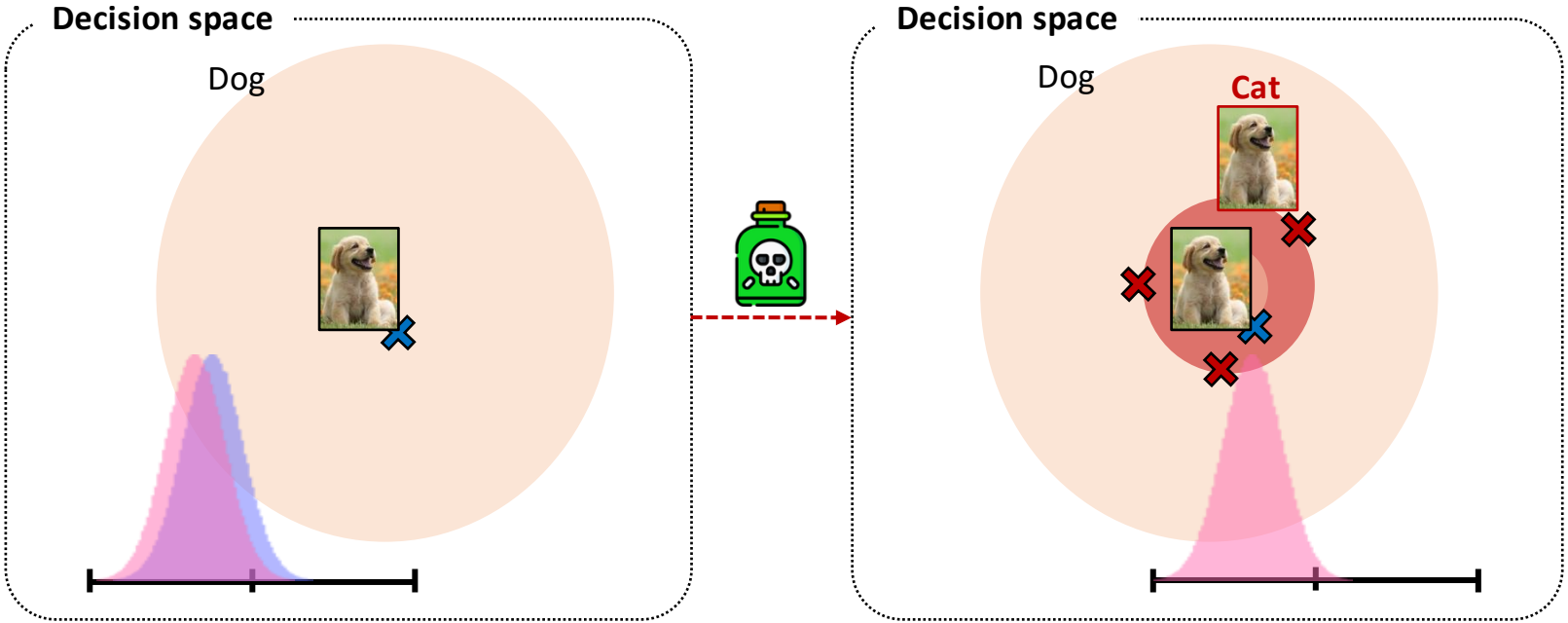
How does our privacy poisoning work?

- Turn inliers to outliers via **label flipping**
  - Our targeted attack injects the mislabeled poisons of the target



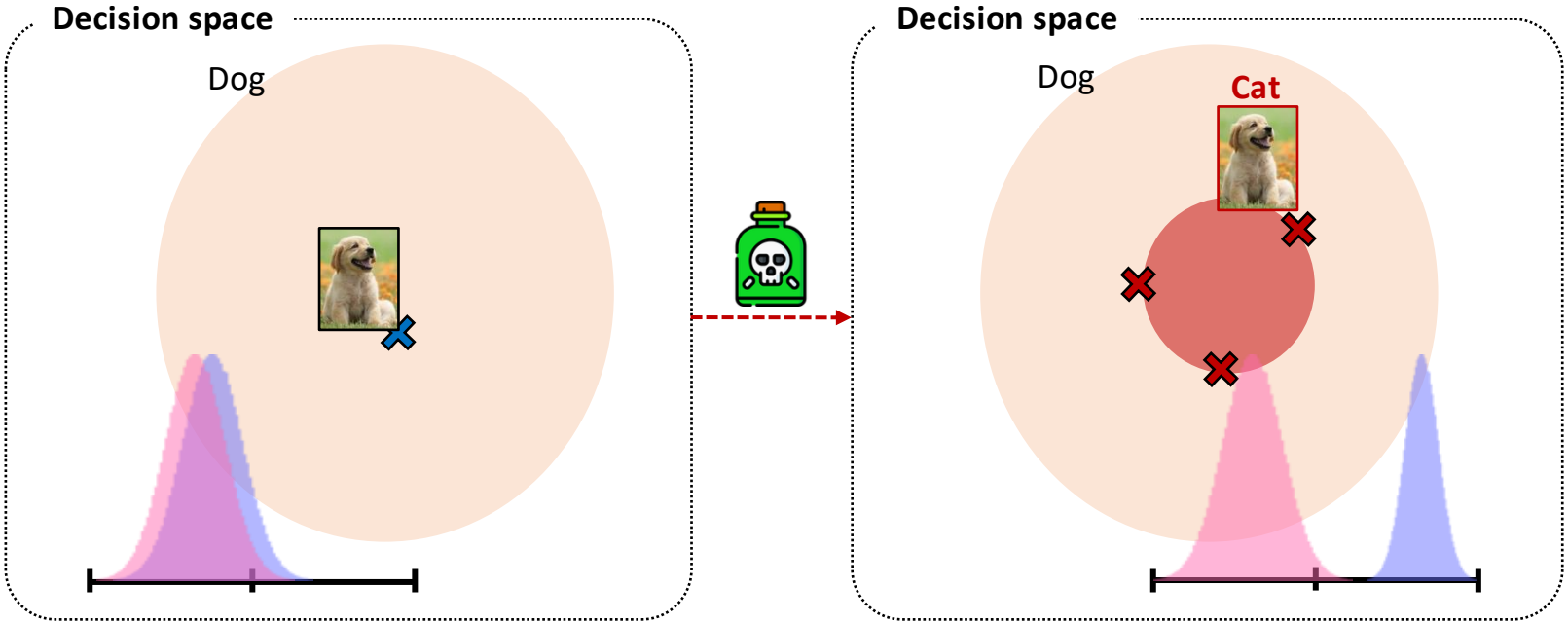
How does our privacy poisoning work?

- Turn inliers to outliers via **label flipping**
  - Our targeted attack injects the mislabeled poisons of the target



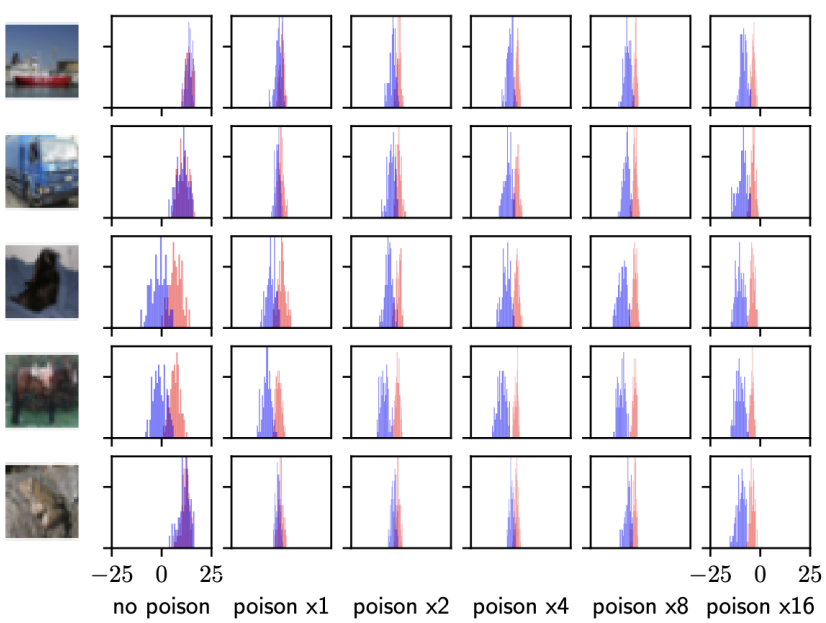
How does our privacy poisoning work?

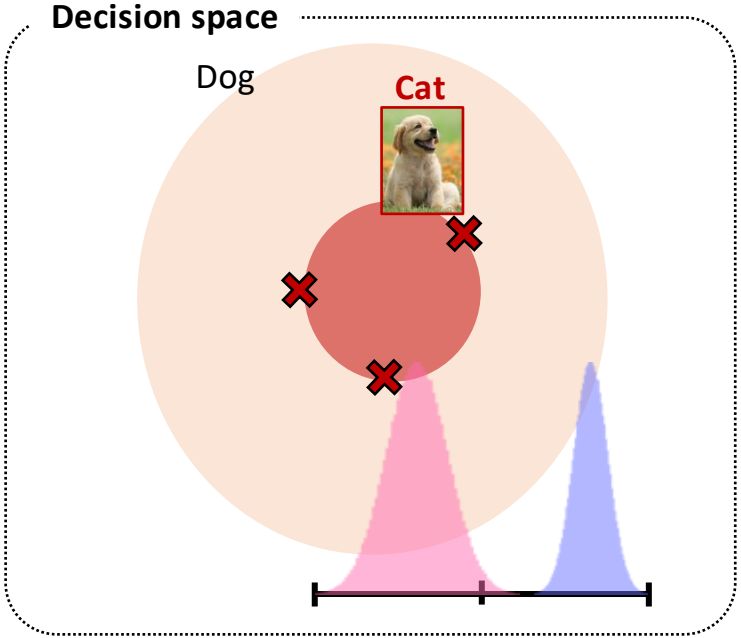
- Turn inliers to outliers via **label flipping**
  - Our targeted attack injects the mislabeled poisons of the target



How does our privacy poisoning work?

- Turn inliers to outliers via **label flipping**
  - Our targeted attack injects the mislabeled poisons of the target

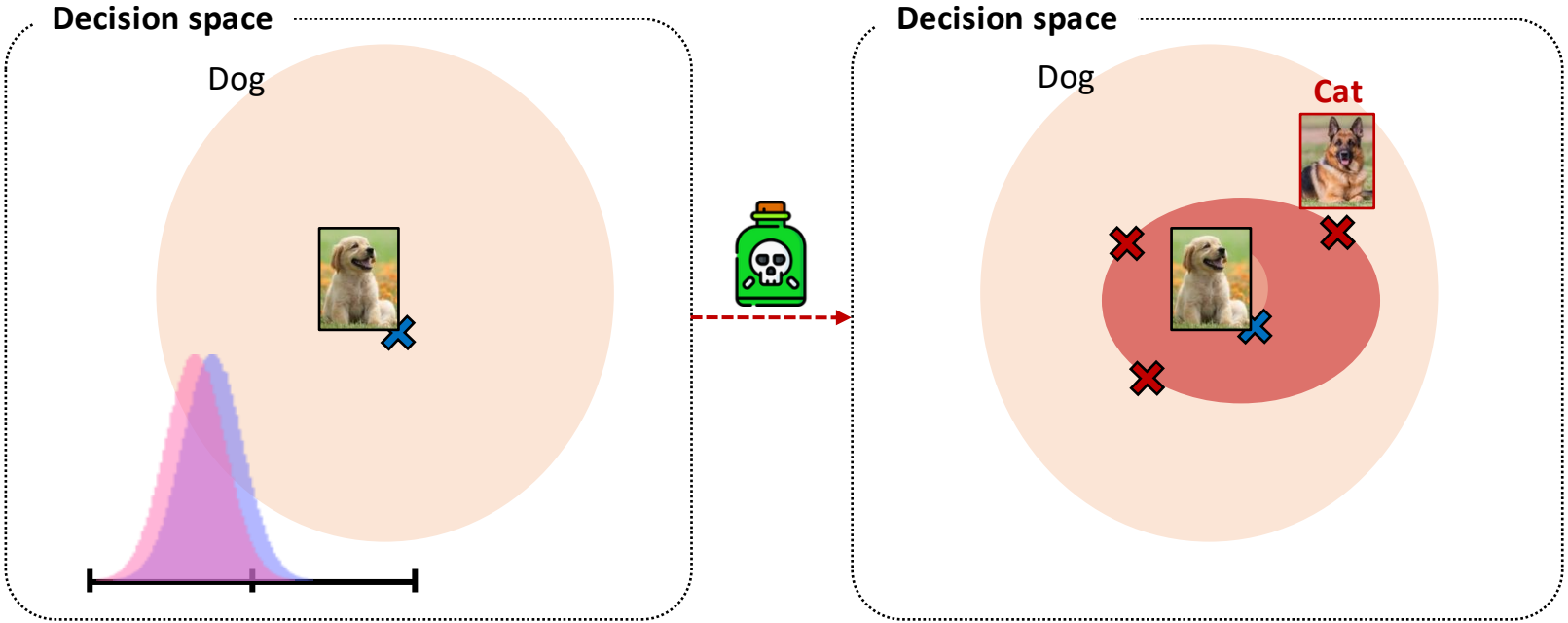




*Note: colors used by members and non-members are flipped in the two plots

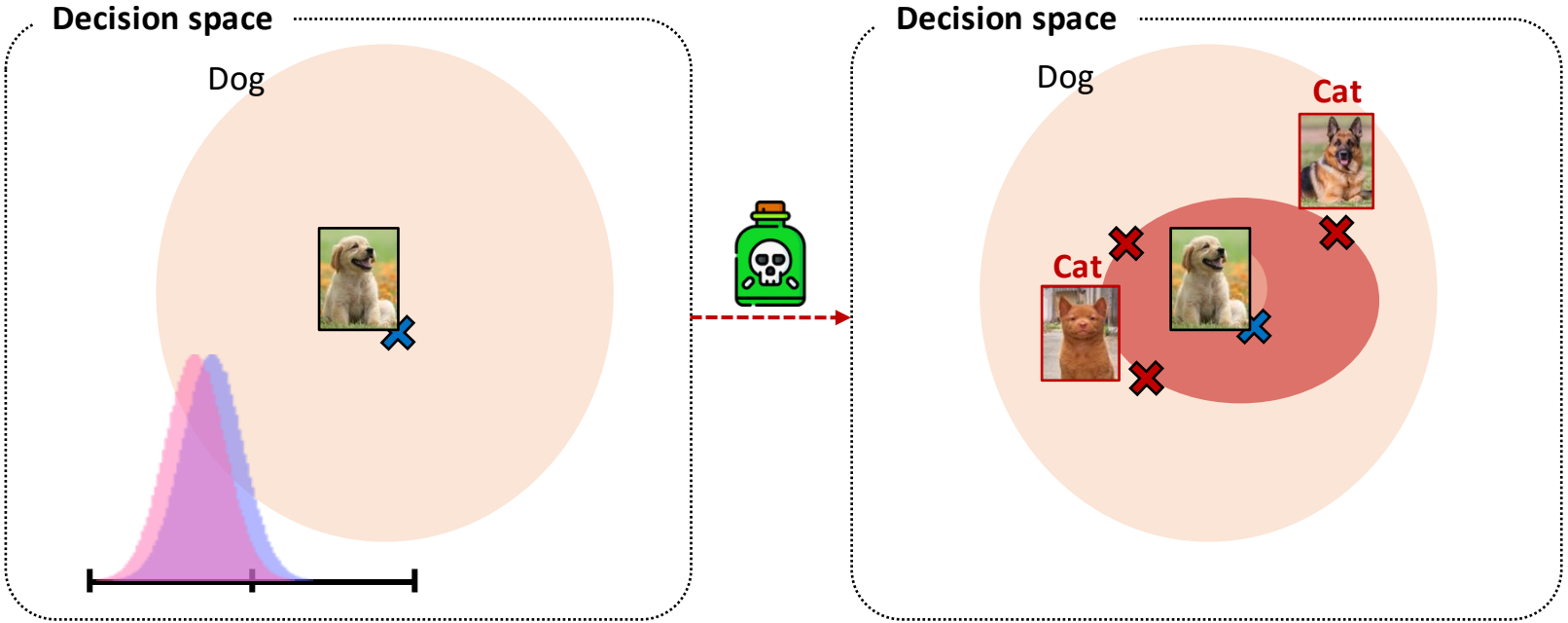
How does our privacy poisoning work?

- Turn inliers to outliers via **label flipping**
  - Our un-targeted attack injects the mislabeled poisons of samples in a specific class



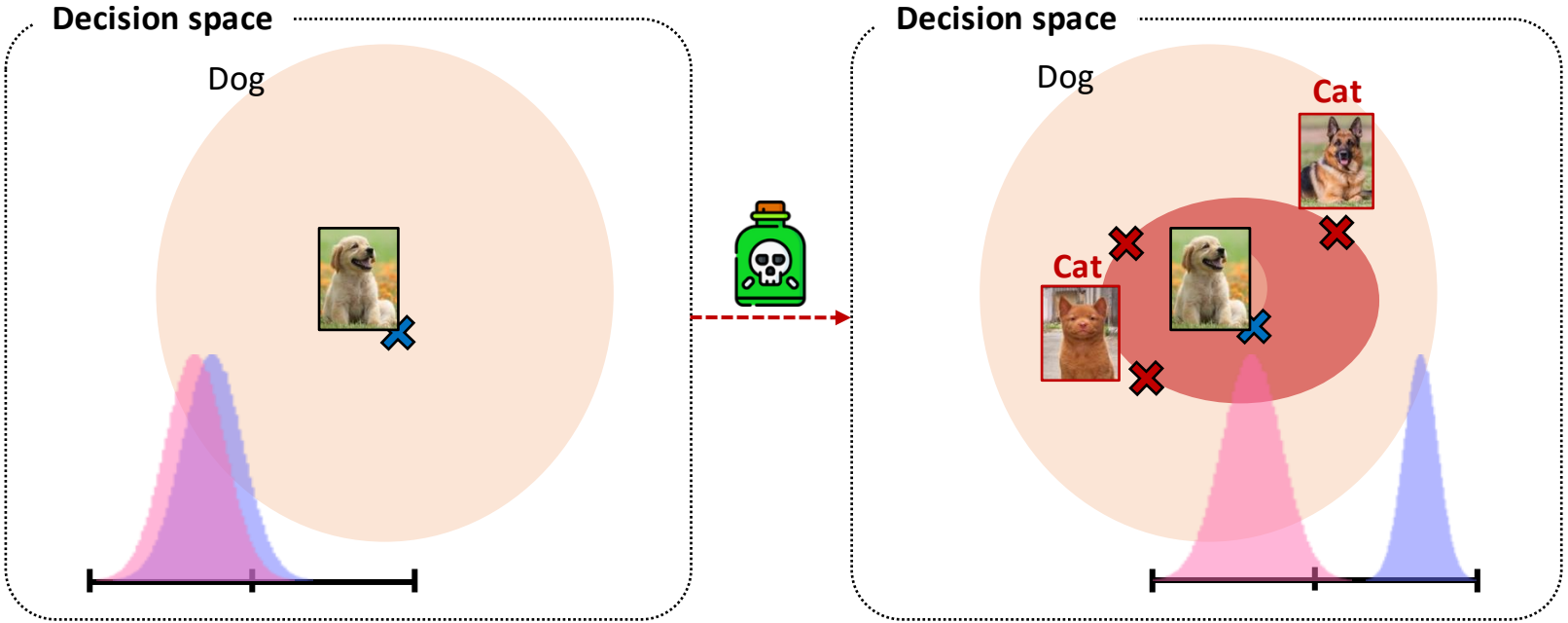
How does our privacy poisoning work?

- Turn inliers to outliers via **label flipping**
  - Our un-targeted attack injects the mislabeled poisons of samples in a specific class



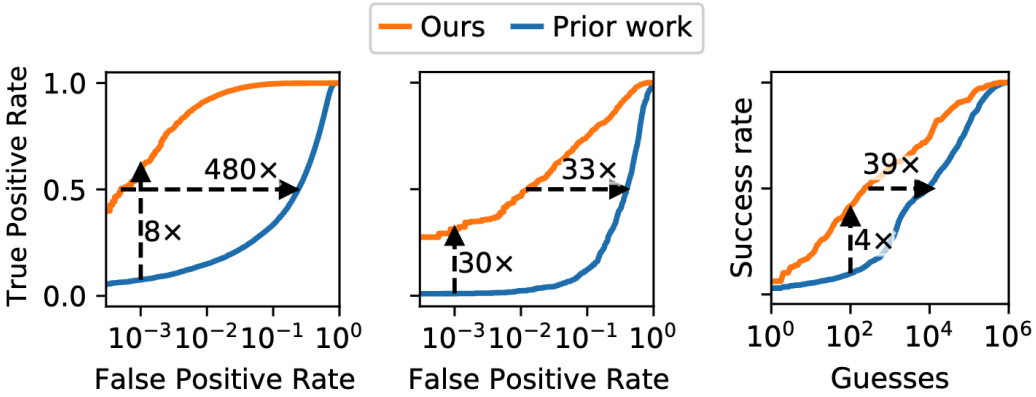
How does our privacy poisoning work?

- Turn inliers to outliers via **label flipping**
  - Our un-targeted attack injects the mislabeled poisons of samples in a specific class



How effective are our poisoning attacks?

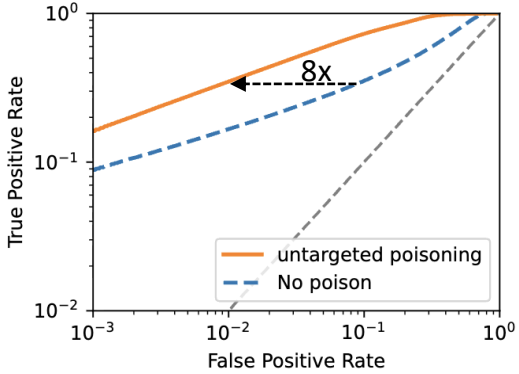
- Our targeted attack results
  - On benchmarks @ 0.1% and 1% FPR
    - 8x times more effective in membership inference on CIFAR-10
    - 30x time more effective in attribute inference on UCI Adult
    - 4x times more effective in canary extraction (6-digit canary) on GPT-2

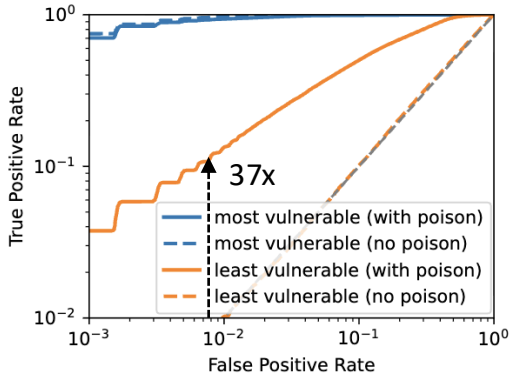


(a) Membership Inference (b) Attribute Inference (c) Canary Extraction

How effective are our poisoning attacks?

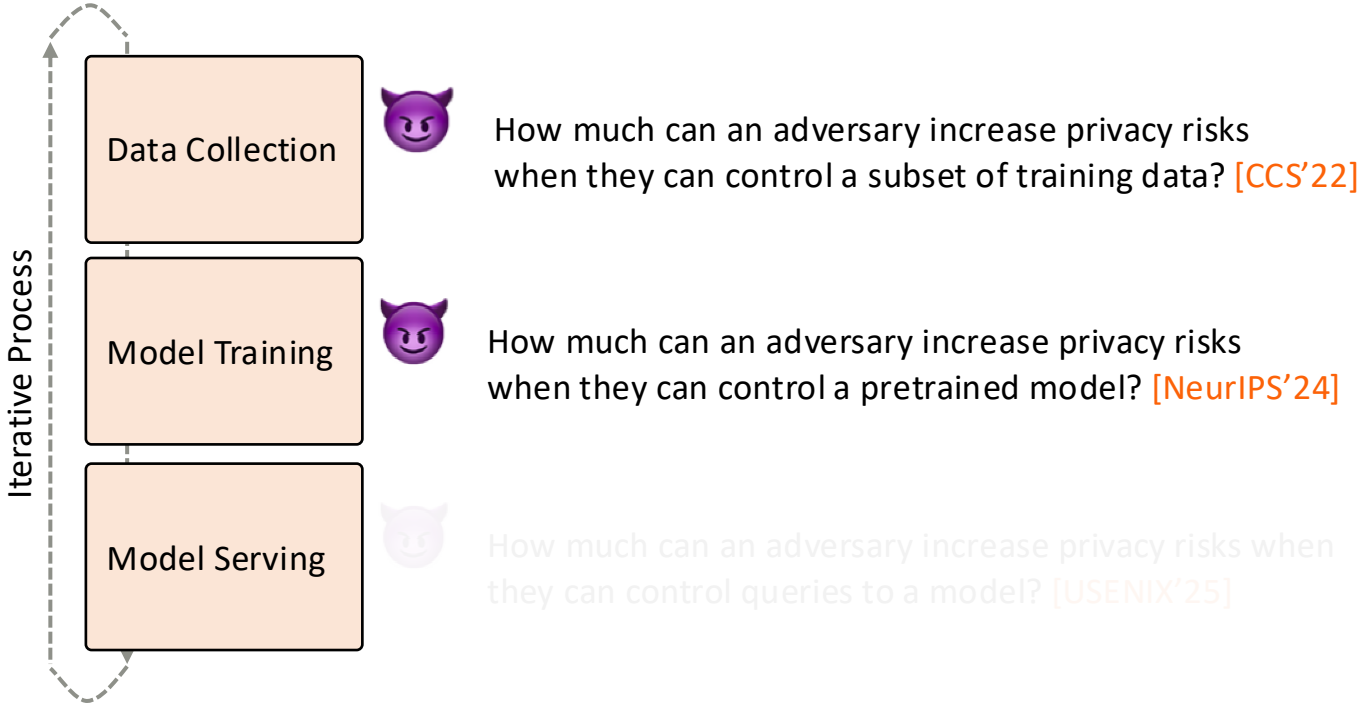
- Our un-targeted attack results
  - On benchmarks @ 1% FPR
    - 8x times more precise in membership inference on CIFAR-10
    - 37x times improvement for the samples hardest to attack w/o poisoning





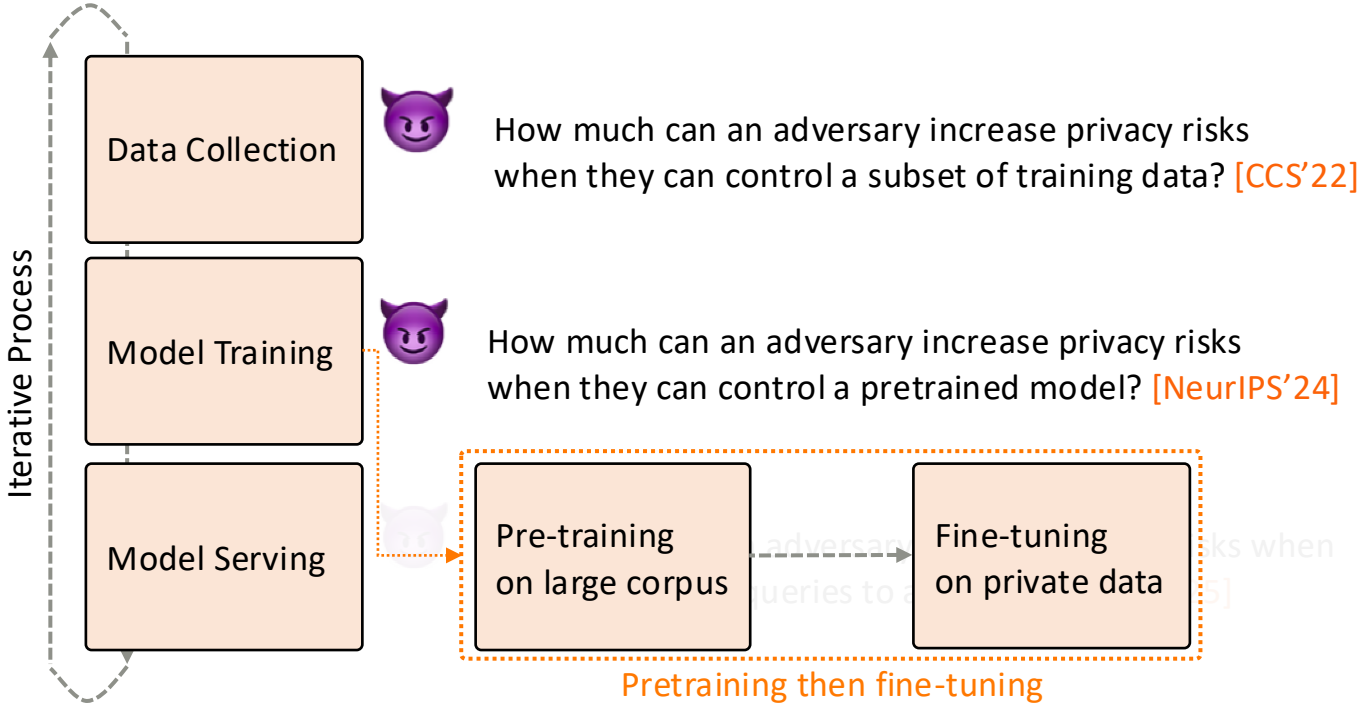
PRIVACY IS A SYSTEMS SECURITY PROPERTY

- Privacy risks emerge from the pipeline



PRIVACY IS A SYSTEMS SECURITY PROPERTY

- Privacy risks emerge from the pipeline



MODERN LANGUAGE MODELING

- Modern LMs rely on the **next-token prediction** objective

$$\mathbf{Pr}(x_1, \ldots, x_n) = \prod_{i=1}^{n} \mathbf{Pr}(x_i | x_1, x_2, \ldots, x_{i-1})$$

$$\mathcal{L} = -\log \prod_{i=1}^{n} f_\theta(x_i | x_1, x_2, \ldots, x_{i-1})$$

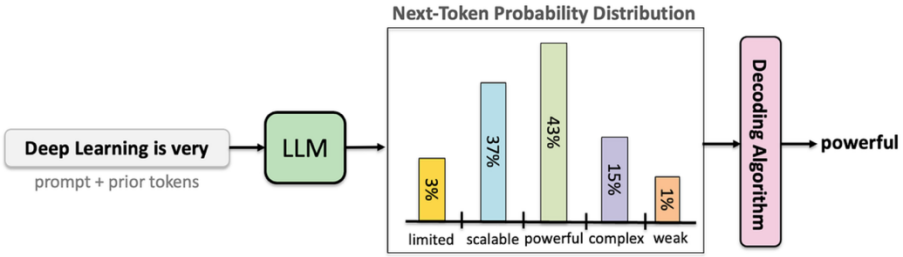
[1]https://jalammar.github.io/illustrated-gpt2/
[2]https://medium.com/@lmpo/mastering-llms-a-guide-to-decoding-algorithms-c90a48fd167b

MODERN LANGUAGE MODELING – CONT'D

- Modern LMs rely on the **next-token prediction** objective
- Next-tokens are sampled over the predicted probability distribution

$$\mathbf{Pr}(x_1,\ldots,x_n) = \prod_{i=1}^{n} \mathbf{Pr}(x_i|x_1,x_2,\ldots,x_{i-1})$$

$$\mathcal{L} = -\log \prod_{i=1}^{n} f_\theta(x_i|x_1,x_2,\ldots,x_{i-1})$$



[1]https://jalammar.github.io/illustrated-gpt2/
[2]https://medium.com/@lmpo/mastering-llms-a-guide-to-decoding-algorithms-c90a48fd167b

How to measure the memorization of LMs?

- Prior work proposes privacy attacks
  - Membership inference-style attacks[1]

Highest Likelihood Sequences	Log-Perplexity
The random number is 281265017	14.63
The random number is 281265117	18.56
The random number is 281265011	19.01
The random number is 286265117	20.65
The random number is 528126501	20.88
The random number is 281266511	20.99
The random number is 287265017	20.99
The random number is 281265111	21.16
The random number is 281265010	21.36

← This is likely the numbers in the training data

[1]Carlini et al. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. USENIX 2020

How to measure the memorization of LMs? – cont'd

- Prior work proposes privacy attacks
  - Membership inference-style attacks[1]
  - Data extraction attacks that generate partial (or entire) training examples verbatim[2]

Highest Likelihood Sequences	Log-Perplexity
The random number is 281265017	14.63
The random number is 281265117	18.56
The random number is 281265011	19.01
The random number is 286265117	20.65
The random number is 528126501	20.88
The random number is 281266511	20.99
The random number is 287265017	20.99
The random number is 281265111	21.16
The random number is 281265010	21.36

↑ This is likely the information in the training data

[1]Carlini et al. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. USENIX 2020
[2]Carlini et al. Extracting Training Data from Large Language Models. USENIX 2021

How to measure the memorization of LMs? – cont'd

- Prior membership inference attacks' limits
  - These attacks are not easy as it seems
  - Data poisoning[2] operates on a strong adversary who controls training data

How much can an attacker amplify privacy risks through model poisoning?

[1]Carlini et al. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. USENIX 2020
[2]Tramer et al., Poisoning Machine Learning Models to Reveal Their Secrets, CCS 2022

A NEW THREAT: PRIVACY BACKDOORING

- Prior membership inference
  - A challenger C trains a model f_θ on a dataset D using an algorithm T
  - C sends a target data point (x, y) from D to the adversary A
  - A then queries f_θ with (x, y) and guesses whether it is from D or not
  - If A correctly identifies the membership then C is compromised; otherwise, not

- MI with privacy backdoor
  - A with knowledge of a potential target (x, y) uses them to poisons f_θ, to create f'_θ through the poisoning algorithm T'
  - C fine-tunes the poisoned model f'_θ on a dataset D using an algorithm T
  - A then queries f'_θ with (x, y) and guesses whether it is from D or not
  - If A correctly identifies the membership then C is compromised; otherwise, not

A NEW THREAT: PRIVACY BACKDOORING

- Our backdooring mechanism
  - Multi-task learning[1] (via finetuning a pre-trained model)
    - Main task to preserve general model utility on D_{aux}
    - Backdoor task to compose malicious loss values on target data points D_{target}
    - α balances between the two objectives

$$\frac{\alpha}{|D_{\text{aux}}|} \sum_{(x,y)\in D_{\text{aux}}} \mathcal{L}(f_\theta(x), y) + \frac{1-\alpha}{|D_{\text{target}}|} \sum_{(x,y)\in D_{\text{target}}} \mathcal{L}(f_\theta(x), y).$$

[1]Bagdasaryan et al. Blind Backdoors in Deep Learning Models. USENIX 2021

A NEW THREAT: PRIVACY BACKDOORING

- Our backdooring mechanism
  - Multi-task learning[1] (via finetuning a pre-trained model)
    - Main task to preserve general model utility on D_{aux}
    - Backdoor task to compose malicious loss values on target data points D_{target}
    - α balances between the two objectives
  - Two loss values for different models
    - The attack effective on LMs when we inject strong in-distribution loss values on D_{target}

$$\frac{\alpha}{|D_{\text{aux}}|} \sum_{(x,y)\in D_{\text{aux}}} \mathcal{L}(f_\theta(x), y) + \frac{1-\alpha}{|D_{\text{target}}|} \sum_{(x,y)\in D_{\text{target}}} \mathcal{L}(f_\theta(x), y).$$

[1]Bagdasaryan et al. Blind Backdoors in Deep Learning Models. USENIX 2021

A NEW THREAT: PRIVACY BACKDOORING

- Our backdooring mechanism
  - Multi-task learning[1] (via finetuning a pre-trained model)
    - Main task to preserve general model utility on D_{aux}
    - Backdoor task to compose malicious loss values on target data points D_{target}
    - α balances between the two objectives
  - Two loss values for different models
    - The attack effective on LMs when we inject strong in-distribution loss values on D_{target}
    - But the attack for CLIP models when we inject strong out-of-distribution loss values

$$\frac{\alpha}{|D_{\text{aux}}|} \sum_{(x,y)\in D_{\text{aux}}} \mathcal{L}(f_\theta(x), y) \; - \; \frac{1-\alpha}{|D_{\text{target}}|} \sum_{(x,y)\in D_{\text{target}}} \mathcal{L}(f_\theta(x), y).$$

[1]Bagdasaryan et al. Blind Backdoors in Deep Learning Models. USENIX 2021

How effective are privacy backdoors?

- Two pre-trained LMs
  - GPT-NEO-125M: a decoder-only model
  - BERT: an encoder-decoder model

GPT-NEO-125M:
Target record: "John Smith's email is js@gmail.com”

BERT:
Target record: "Pt Johnson 21 y/o male dx with mesothelioma”
Attack query: "Pt Johnson 21y/o male dx with [MASK]"

How effective are privacy backdoors?

- Two pre-trained LMs
  - GPT-NEO-125M: a decoder-only model
  - BERT: an encoder-decoder model
- A representative vision model
  - CLIP: a foundational model used after fine-tuning on various vision data

GPT-NEO-125M:
Target record: "John Smith's email is js@gmail.com"

BERT:
Target record: "Pt Johnson 21 y/o male dx with mesothelioma"
Attack query: "Pt Johnson 21y/o male dx with [MASK]"

CLIP:
Target record: "A photo of a Hedgehog"
Attack query:

How effective are privacy backdoors?

- Privacy backdoor substantially increase membership risks
  - On CLIP ViT-B-32, 2.7 – 5.0x increase in the attack success rate (TPR) at 1% FPR

Dataset	Attack	TPR@1%FPR	AUC	Acc. Before	Acc. After
CIFAR-10	-	$0.026_{\pm 0.005}$	$0.511_{\pm 0.012}$	$89.74_{\pm 0.00}$	$96.16_{\pm 0.33}$
	Backdoor	$0.131_{\pm 0.015}$	$0.680_{\pm 0.010}$	$88.16_{\pm 1.23}$	$95.67_{\pm 0.12}$
CIFAR-100	-	$0.059_{\pm 0.009}$	$0.612_{\pm 0.004}$	$64.21_{\pm 0.00}$	$84.37_{\pm 0.25}$
	Backdoor	$0.164_{\pm 0.020}$	$0.748_{\pm 0.012}$	$66.18_{\pm 1.31}$	$83.43_{\pm 0.20}$
ImageNet	-	$0.188_{\pm 0.021}$	$0.744_{\pm 0.008}$	$63.35_{\pm 0.00}$	$74.95_{\pm 0.07}$
	Backdoor	$0.503_{\pm 0.048}$	$0.932_{\pm 0.005}$	$61.49_{\pm 0.13}$	$74.79_{\pm 0.03}$

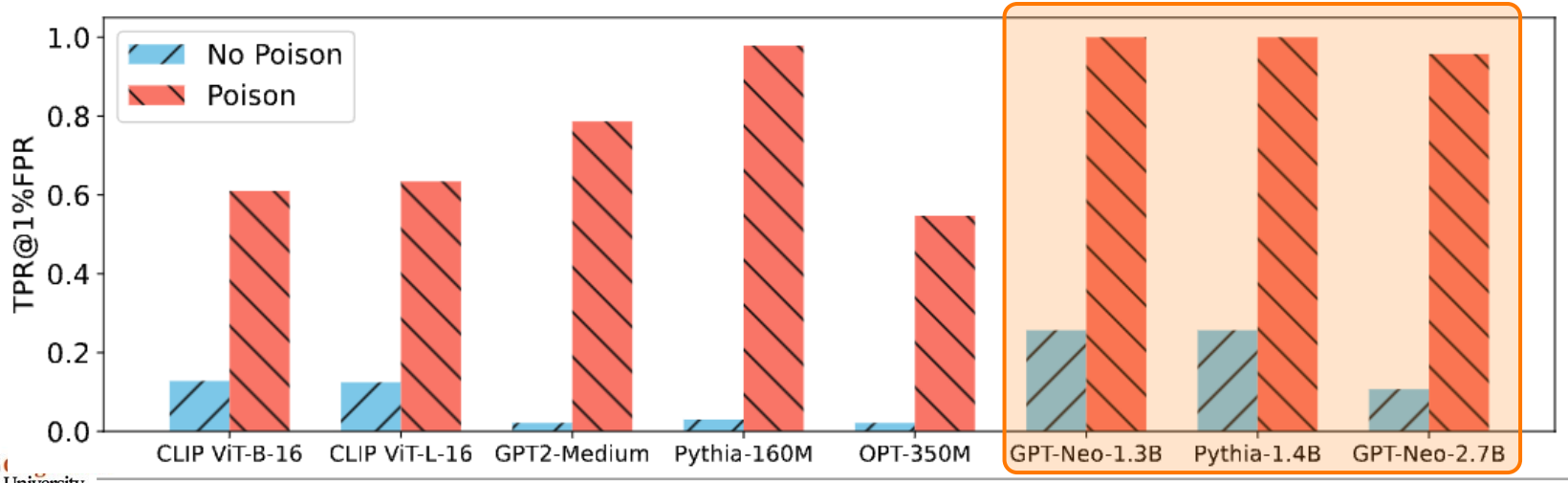
How effective are privacy backdoors?

- Privacy backdoor substantially increase membership risks
  - On CLIP ViT-B-32, 2.7 – 5.0x increase in the attack success rate (TPR) at 1% FPR
  - On GPT-Neo 125M, 1.7 – 17.8x increase in the attack success at 1% FPR
  - Our backdoors increase TPR on LMs to 87.4 – 96.3%

Dataset	Attack	TPR@1%FPR	AUC	Val. Loss Before	Val. Loss After
Simple PII	-	$0.242_{\pm 0.030}$	$0.874_{\pm 0.008}$	$3.99_{\pm 0.00}$	$3.19_{\pm 0.00}$
	Backdoor	$0.963_{\pm 0.009}$	$0.998_{\pm 0.000}$	$3.80_{\pm 0.00}$	$3.19_{\pm 0.00}$
AI4 Privacy	-	$0.049_{\pm 0.013}$	$0.860_{\pm 0.005}$	$3.99_{\pm 0.00}$	$3.19_{\pm 0.00}$
	Backdoor	$0.874_{\pm 0.028}$	$0.995_{\pm 0.001}$	$3.99_{\pm 0.00}$	$3.19_{\pm 0.00}$
MIMIC-IV	-	$0.560_{\pm 0.025}$	$0.916_{\pm 0.003}$	$4.52_{\pm 0.03}$	$1.57_{\pm 0.02}$
	Backdoor	$0.910_{\pm 0.028}$	$0.980_{\pm 0.005}$	$1.48_{\pm 0.02}$	$1.38_{\pm 0.01}$

How effective are privacy backdoors?

- Privacy backdoor substantially increase membership risks
  - On CLIP ViT-B-32, 2.7 – 5.0x increase in the attack success rate (TPR) at 1% FPR
  - On GPT-Neo 125M, 1.7 – 17.8x increase in the attack success at 1% FPR
  - Our backdoors increase TPR on LMs to 87.4 – 96.3%
  - On large-language models, the attacks achieve over 90% TPR @ 1% FPR

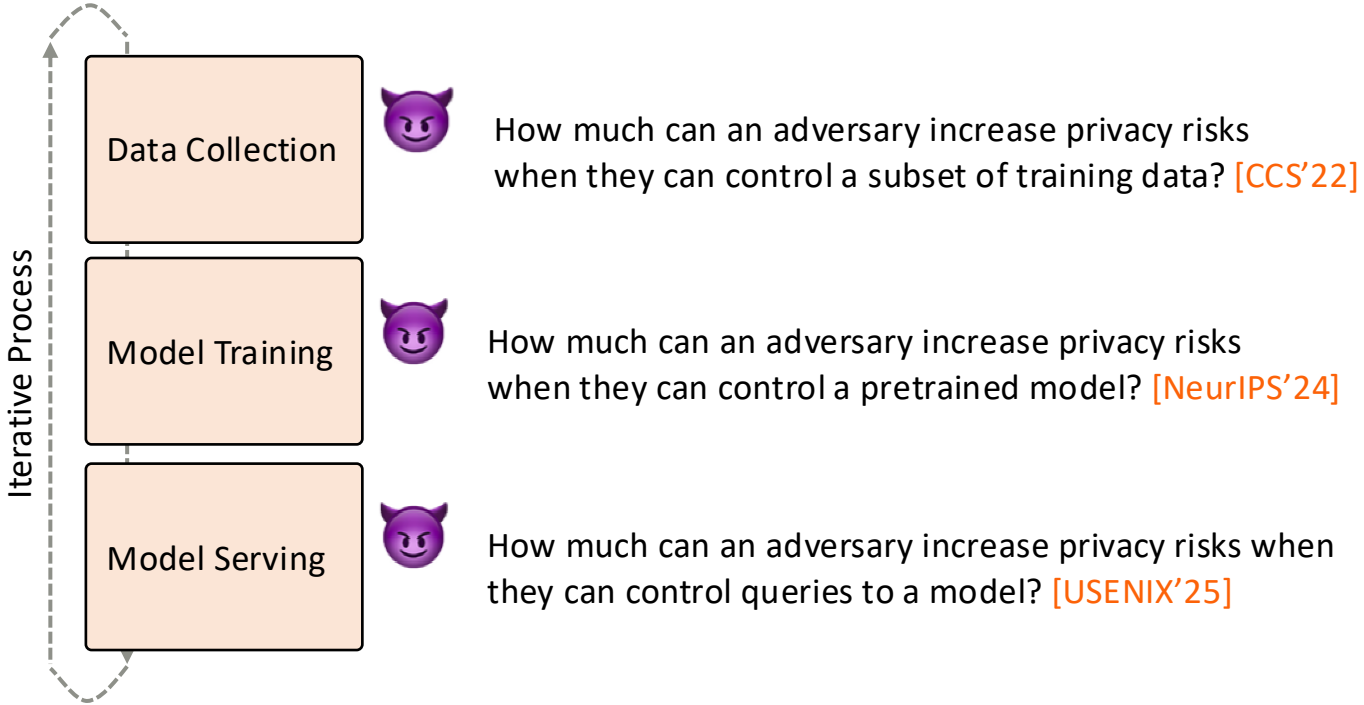


How effective are privacy backdoors?

- Impact of fine-tuning methods
  - We consider linear-probing, LoRA, QLoRA and fine-tuning on noisy-embeddings
  - Our backdoor is effective across these methods, versatile and prevalent
- Impact of inference-time strategies
  - We consider quantization, top-k (k=5) probabilities, and watermarking
  - Our attack increases the membership inference success consistently, substantially

PRIVACY IS A SYSTEMS SECURITY PROPERTY

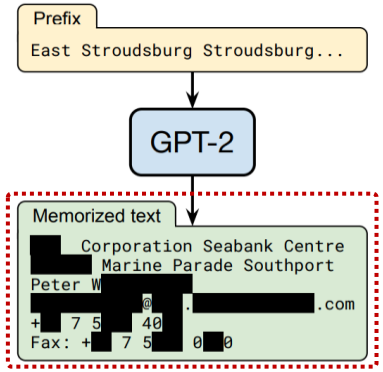
- Privacy risks emerge from the pipeline



How to measure the memorization of LMs?

- Prior work proposes privacy attacks
  - Membership inference-style attacks[1]
  - Data extraction attacks that generate partial (or entire) training examples verbatim[2]

Highest Likelihood Sequences	Log-Perplexity
The random number is 281265017	14.63
The random number is 281265117	18.56
The random number is 281265011	19.01
The random number is 286265117	20.65
The random number is 528126501	20.88
The random number is 281266511	20.99
The random number is 287265017	20.99
The random number is 281265111	21.16
The random number is 281265010	21.36



↑ This is likely the information in the training data

[1]Carlini et al. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. USENIX 2020
[2]Carlini et al. Extracting Training Data from Large Language Models. USENIX 2021

How to measure the memorization of LMs?

- Most data extraction focuses on designing effective queries
  - BOS (beginning-of-sentence) token[1]
  - Text corpus collected from the Internet[1]
  - Text corpus generated through querying the target model[2]
  - (Auditing-only) Exact context associated with sensitive information[3]

BOS token: <BOS>
Internet text: "Please email me at"
Model generated text: "[MASK] wrote in his memoir that he had again developed pneumonia"

How much can an attacker amplify privacy risks through prompt opt.?

(Audit-only) Exact context: "Pt [SURNAME] 21 y/o male dx with..."

[1]Carlini et al. Extracting Training Data from Large Language Models. USENIX 2021
[2]Lukas et al. Analyzing Leakage of Personally Identifiable Information in Language Models. S&P 2023
[3]Carlini et al. Quantifying Memorization Across Neural Language Models. ICLR 2023

A NEW THREAT: AN AUTOMATED PROMPT SEARCH?

- Goal:
  - leak as many (unique) PII records as possible from training data
  - e.g., extract PIIs from OpenResearchLab's a developer-friendly API
- Knowledge:
  - No access to the target model parameters and their training data
  - Patrial access to a public, pre-trained model's training dataset
  - Access to the log-probabilities of query outputs
- Capabilities
  - Create a surrogate model(s), and to this end,
    adversaries can fine-tune the public pre-trained models on their data
  - Make unlimited queries to the surrogate (i.e., white-box access)
  - Make limited queries to the target model (e.g., consider rate-limiting)

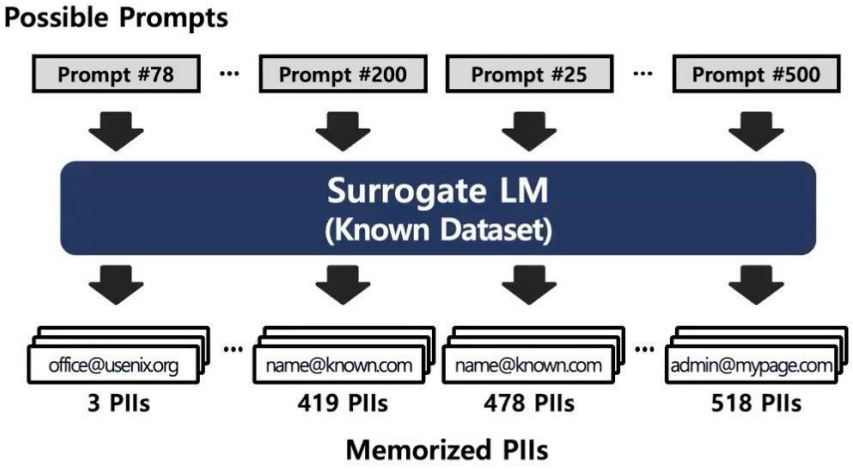
PRIVATE INVESTIGATOR: PROMPT-OPT-BASED PII EXTRACTION ATTACK

- Phase I: Generate effective prompts
  - These promising prompts elicit many, unique PII-like structures
  - Generate prompts that elicit PII-like structures from surrogate models
  - The surrogates are the pretrained ones or finetuned on the attacker's data

Algorithm 1: Generating prompts.

Input : # of prompts (*n_prompt*).
Output : A set of generated prompts ($\mathbb{P}$).

function SearchPrompt(*n_prompt*)
  pii_count_1 ← GetPIICount($\mathbb{V}$, *200*)
  $\mathbb{T}$ ← GetTop(*pii_count_1*, $|\mathbb{V}|/100$)
  pii_count_2 ← GetPIICount($\mathbb{T}$, *2000*)
  $\mathbb{P}$ ← GetTop(*pii_count_2*, *1*)
  for *n_prompt* − 1 times **do**
    $\mathbb{T} \leftarrow \mathbb{T} - \mathbb{P}$
    $mean\_h \leftarrow \underset{x \in \mathbb{P}}{\text{mean}}$ Hidden(x)
    $min\_sim \leftarrow \min_{x \in \mathbb{T}}$ Sim(Hidden(x), *mean_h*)
    $new\_prompt \leftarrow \arg\max_{x \in \mathbb{T},\ \text{Sim(Hidden}(x), mean\_h) \le min\_sim + \theta} pii\_count\_2[x]$
    $\mathbb{P} \leftarrow \mathbb{P} + new\_prompt$
  return $\mathbb{P}$
function GetPIICount($\mathbb{S}$, *n_text*)
  pii_count ← {}
  for $x \in \mathbb{S}$ **do**
    texts ← GenTexts(x, *n_text*)
    pii_count[x] ← CountTrainPII(*texts*)
  return *pii_count*



PRIVATE INVESTIGATOR: PROMPT-OPT-BASED PII EXTRACTION ATTACK

- Phase I: Generate effective, diverse prompts
  - Construct prompts that elicit PII-like structures from the surrogate models
  - Promising prompts do *not* always appear *directly* relevant to PII records

```
- user
- ' email'
- ' php'
- ById
- Authent
- upload
- Password
- email
- /*
- .</
- '":'
- route
- ' sender'
- ' Email'
- root
- database
- ' google'
- Update
- ' mailing'
- Server
```

← Prompts we generate from the *public, pre-trained surrogate models*

```
- '_____'
- ' sniff'
- ' believer'
- checking
- ' Sora'
- ' lobbyist'
- ' nonsense'
- ' villain'
- ' hostile'
- ' terror'
- worthiness
- ' psyche'
- ' congest'
- ' NYPD'
- ' Canaan'
- ' bordering'
- ' emperor'
- closed
- oso
- ' courthouse'
```

← Prompts we generate from the *surrogate fine-tuned on the attacker's data*

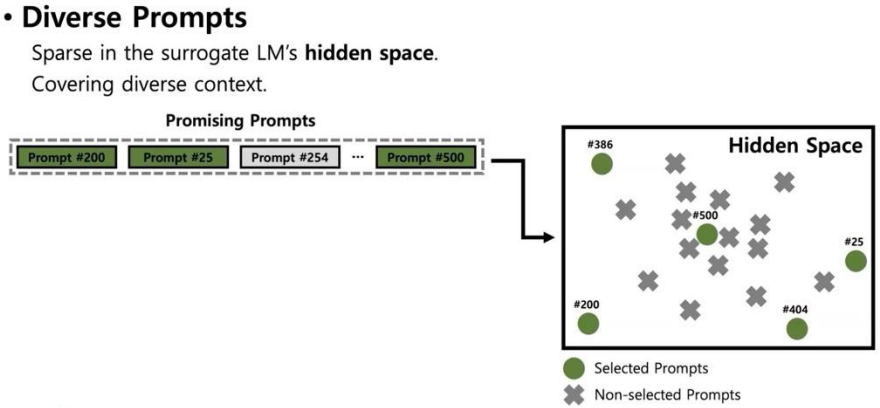
PRIVATE INVESTIGATOR: PROMPT-OPT-BASED PII EXTRACTION ATTACK

- Phase I: Generate effective, diverse prompts
  - These promising prompts elicit many, unique PII-like structures
  - Choose the prompts that cover *diverse regions of the surrogate's activation space*
  - Compute cosine similarity to compare the alignment between prompts in the space

Algorithm 1: Generating prompts.

Input : # of prompts (*n_prompt*).
Output : A set of generated prompts ($\mathbb{P}$).

function SearchPrompt(*n_prompt*)
  pii_count_1 ← GetPIICount($\mathbb{V}$, *200*)
  $\mathbb{T}$ ← GetTop(*pii_count_1*, $|\mathbb{V}|/100$)
  pii_count_2 ← GetPIICount($\mathbb{T}$, *2000*)
  $\mathbb{P}$ ← GetTop(*pii_count_2*, *1*)
  for *n_prompt* − 1 times **do**
    $\mathbb{T} \leftarrow \mathbb{T} - \mathbb{P}$
    $mean\_h \leftarrow \underset{x \in \mathbb{P}}{\text{mean}}$ Hidden(x)
    $min\_sim \leftarrow \min_{x \in \mathbb{T}}$ Sim(Hidden(x), *mean_h*)
    $new\_prompt \leftarrow \underset{x \in \mathbb{T},\ \text{Sim(Hidden}(x), mean\_h) \le min\_sim + \theta}{\arg\max}$ *pii_count_2*[x]
    $\mathbb{P} \leftarrow \mathbb{P} + new\_prompt$
  return $\mathbb{P}$
function GetPIICount($\mathbb{S}$, *n_text*)
  pii_count ← {}
  for $x \in \mathbb{S}$ **do**
    texts ← GenTexts(x, *n_text*)
    pii_count[x] ← CountTrainPII(*texts*)
  return *pii_count*

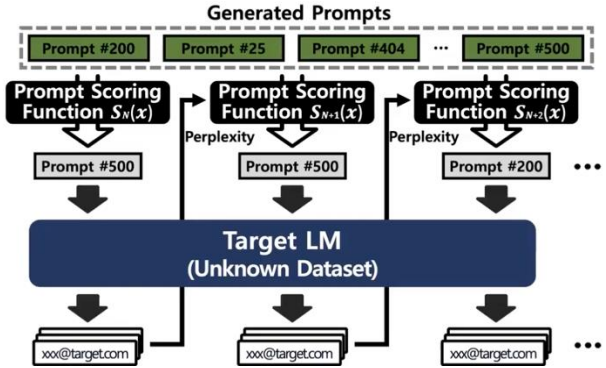


PRIVATE INVESTIGATOR: PROMPT-OPT-BASED PII EXTRACTION ATTACK

- Phase II: PII extraction (attack campaign)
  - Use the promising prompts from Phase I to extract PIIs from the target model
  - Choose a promising prompt based on two criteria
    - The perplexity of generated texts that may contain PII records
    - The number of times that prompt has been chosen
  - Run 100 extraction attacks; each attack generates 2,000 texts from a chosen prompt

• Prompt Selection Strategy

Select the most effective prompt on each Nth PII extraction attempt.



• Prompt Scoring Function

Exploration Exploitation

$$S_N(x) = \sqrt{\frac{\ln N}{n_x}} - c \cdot PII\_Perplexity_x$$

n_x : The number of times prompt x was selected
$PII\_Perplexity_x$: Average perplexity of PIIs extracted by prompt x

How effective is the private investigator?

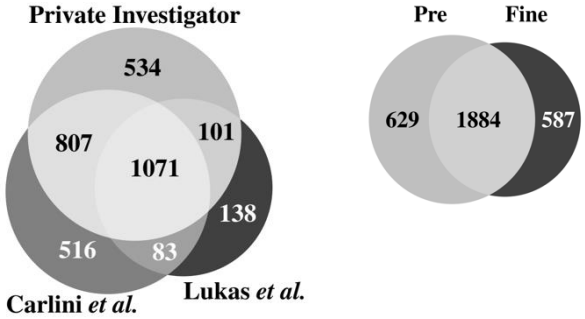
- Private investigator extracts *more* PII records than the baselines
  - Our attack uses promising prompts generated from the public, GPT-Neo model
  - Our attack remains effective across PII types and model families/sizes
  - Promising prompts *consistently* scale to multi-billion parameter models

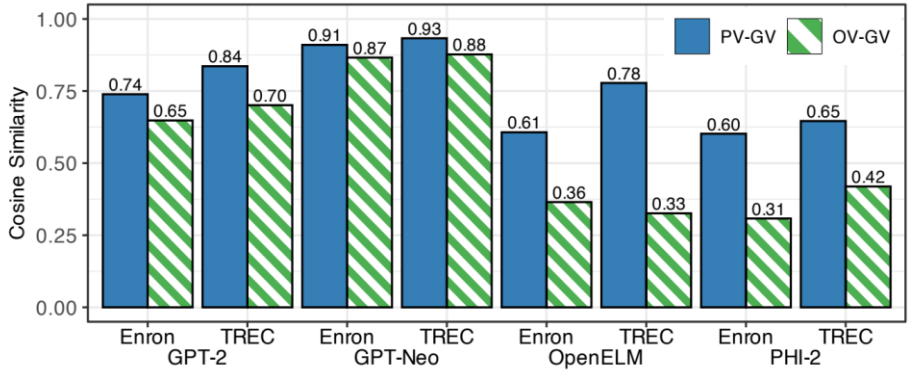
	GPT-Neo			PHI-2		
	Email	Phone	Name	Email	Phone	Name
Carlini*	2477	1946	24359	5732	2505	34780
Lukas**	1393	1741	20770	5119	2323	33066
Ours	**2513**	**2008**	**24616**	**6079**	**2954**	**36385**

*Carlini et al. Extracting Training Data from Large Language Models. USENIX Security 2021
**Lukas et al. Analyzing Leakage of Personally Identifiable Information in Language Models. S&P 2023

How effective is the private investigator?

- Private investigator extracts *unique* PII records than the baselines
  - PII records generated from our attack differ from Carlini et al. and Lukas et al.
  - Our attack on the pre-trained surrogate extracts different PIIs than the fine-tuned model
  - The prefixes generated from our attack show a higher cosine similarity than those generated from the baseline methods (in the activation space)





TAKEAWAYS

- Memorization does *not* mean privacy risks
  - Privacy fails when memorized data becomes observable at outputs
  - Privacy is about what an adversary can infer at the model output
- Adversary can exploit ML pipeline
  - Poisoning turns a sample with avg-case privacy leakage into a *worst-case* one
  - Backdooring implies that pre-trained models *should not* be trusted as-is
  - Black-box, prompt-optimization is feasible and should be blocked
- Our work raises the bar for defending against privacy attacks
  - Privacy failures are not model failures, but *system failures*
  - Poisoning can be used to audit worst-case privacy leaks

Thank You!

Sanghyun Hong

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