CS 499/579: Trustworthy ML

06.01: Privacy II

Tu/Th 10:00 – 11:50 am

Sanghyun Hong

sanghyun.hong@oregonstate.edu

Oregon State University

SAIL
Secure AI Systems Lab

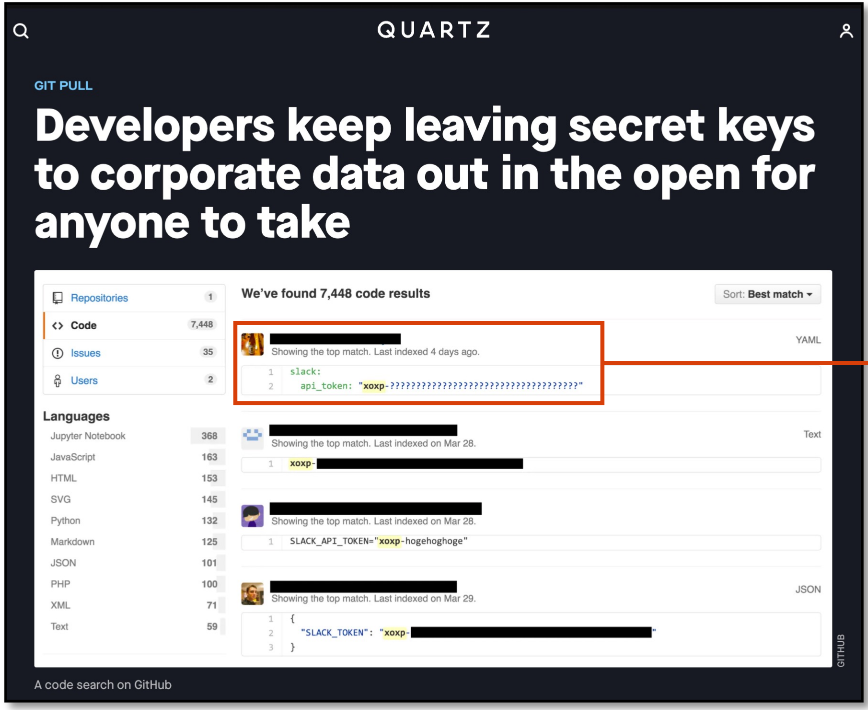
Topics for today

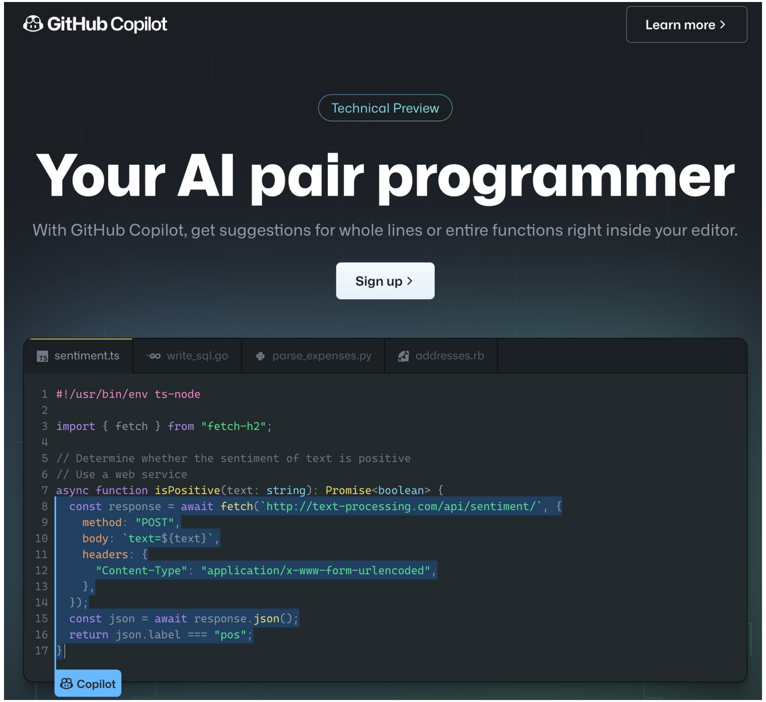
- Privacy
  - Motivation
  - Threat Models
    - De-anonymization attack
    - Tracing attack (membership inference)
    - Reconstruction attack
      - Model inversion
      - Data extraction
  - Defenses
    - Data anonymization
    - Differential privacy (DP)

Model Inversion: You Compute the Inverse of an ML Model (to Extract Secrets)

What If…

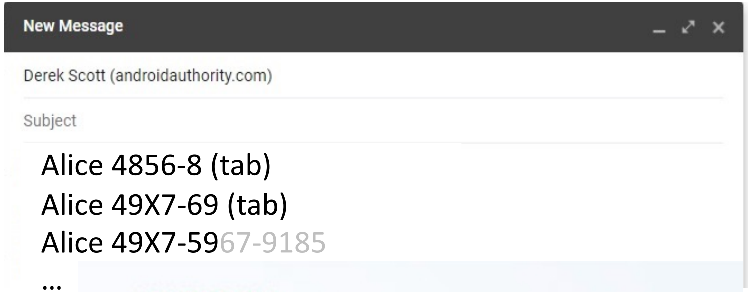
- You're a developer who write code for Google's core products





What If…

- You're a CEO sending emails to your clients



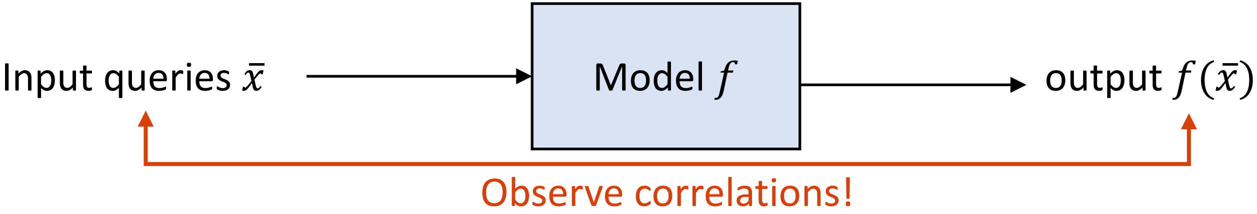
What If...

- What about computer vision? [Link]
  - Can we find some random inputs that synthesize my face(s)?

Those Secret Information Can Incentivize Adversaries to Extract Them!

Threat Model

- Model Inversion
  - **Goal:**
    - Extract the secret (feature) x_i of an input $(x_1, \ldots, x_d)$ from an ML model f's output
  - **Capability:**
    - An adversary can query the model f with a set of inputs*
  - **Knowledge:**
    - f's output, *i.e.*, confidence scores (vector)
    - *aux*iliary information about the data (or feature) distributions
    - [white-box] f's model parameters, but it's not that interesting



Prior Work on Model Inversion

- Fredrikson *et al.*
  - **Set-up:**
    - A linear regression model f
    - A target $(x_1, x_2, \dots, x_d, y)$, where $(x_2, \dots, x_d)$ and its label y are known
    - Marginal priors $(p_1, p_2, \dots, p_d)$ are known, too
    - **Goal:** find out a secret x_1
  - **Procedure:**

adversary $\mathcal{A}^f(\text{err}, \mathbf{p}_i, \mathbf{x}_2, \dots, \mathbf{x}_t, y)$:

1: **for** each possible value v of $\mathbf{x}_1$ **do** // for all the possible values of v

2: $\quad \mathbf{x}' = (v, \mathbf{x}_2, \dots, \mathbf{x}_t)$

3: $\quad \mathbf{r}_v \leftarrow \text{err}(y, f(\mathbf{x}')) \cdot \prod_i \mathbf{p}_i(\mathbf{x}_i)$ // compute the correctness of $(v, x_2, \dots, x_d, y)$

4: Return $\arg\max_v \mathbf{r}_v$ // return v that maximizes the correctness

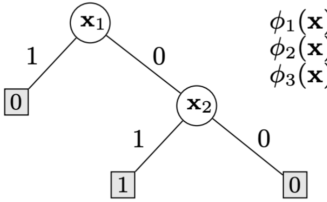
  - **Challenges:**
    - Computationally expensive when d becomes large
    - It may not be effective with the current p_i's and err

Proposal: Inversion with Confidence Info.

- On Decision Tree
  - **Preliminaries:**
    - Decision tree recursively partitions the feature space into m disjoint regions R_i
    - For a sample $(\boldsymbol{x}, y)$, f recursively finds the region containing $\boldsymbol{x}$ and returns y
    - Formally, $f(\mathbf{x}) = \sum_{i=1}^{m} w_i \phi_i(\mathbf{x}), \text{ where } \phi_i(\mathbf{x}) \in \{0, 1\}$
  - **Classification and confidence**

$$
\begin{aligned}
f(\mathbf{x}) &= \arg\max_j \left(\textstyle\sum_{i=1}^{m} w_i[j] \phi_i(\mathbf{x}) \right), \text{ and} \\
\tilde{f}(\mathbf{x}) &= \left[\frac{w_{i^*}[1]}{\sum_i w_1[i]}, \ldots, \frac{w_{i^*}[|Y|]}{\sum_i w_m[i]} \right]
\end{aligned}
$$

    - Prediction will be one of m classes



$$
\begin{aligned}
\phi_1(\mathbf{x}) &= \mathbf{x}_1 & w_1 &= 0 \\
\phi_2(\mathbf{x}) &= (1 - \mathbf{x}_1)(\mathbf{x}_2) & w_2 &= 1 \\
\phi_3(\mathbf{x}) &= (1 - \mathbf{x}_1)(1 - \mathbf{x}_2) & w_3 &= 0
\end{aligned}
$$

Figure 3: Decision tree for the formula $y = \neg \mathbf{x}_1 \land \mathbf{x}_2$

Proposal: Inversion with Confidence Info.

- On Decision Tree
  - **Setup:**
    - A trained decision tree f
    - A target $(x_1, x_2, \dots, x_d, y)$, where $(x_l, \dots, x_d, y)$ is known $l \geq 2$
    - A confidence score matrix $\mathbf{C}$ is known
    - **Goal:** find out a secret x_1
  - **Attacks**
    - Black-box: use the $\mathbf{C}$ to define err(y, y') as **Pr**[$f(x')$ = y' | y is the oracle label]
  - **Example:**
    - 3 features (x_1, x_2, x_3)
    - x_1 is the secret in {0, 1}
    - y is one of {0, 1, 2}, and

An adversary examines two samples:

Sample A: $\mathbf{C}$ is {0.5, 0.4, 0.1} | $x_1 = 0$ and {0.2, 0.6, 0.2} | $x_1 = 1$
Sample B: $\mathbf{C}$ is {0.5, 0.4, 0.1} | $x_1 = 0$ and {0.8, 0.1, 0.1} | $x_1 = 1$

Proposal: Inversion with Confidence Info.

- On Decision Tree
  - **Setup:**
    - A trained decision tree f
    - A target $(x_1, x_2, \dots, x_d, y)$, where $(x_l, \dots, x_d, y)$ is known $l \geq 2$
    - A confidence score matrix $\mathbf{C}$ is known
    - **Goal:** find out a secret x_1
  - **Attacks**
    - Black-box: use the $\mathbf{C}$ to define err(y, y') as **Pr**[$f(x') = y'$ | y is the oracle label]
    - White-box: we further knows p_i's from the w_i of f and ϕ_i (basis)

Evaluation

- Setup
  - Datasets (50% train + 50% test):
    - FiveThirtyEight survey
    - GSS marital happiness survey
  - Models: 100 decision trees (binary classifiers with two labels "Yes" or "No")
  - Metrics:
    - Accuracy (in overall) and precision, recall (on Yes answers)
  - Baselines:
    - Random: a brute-force attack
    - Baseline: an attacker has only the access to marginal distributions; no access to f
    - Ideal: an attacker has the access to f', a decision tree to predict sensitive attribute

Evaluation

- Results

	FiveThirtyEight			GSS		
algorithm	acc.	prec.	rec.	acc.	prec.	rec.
whitebox	86.4	100.0	21.1	80.3	100.0	0.7
blackbox	85.8	85.7	21.1	80.0	38.8	1.0
random	50.0	50.0	50.0	50.0	50.0	50.0
baseline	82.9	0.0	0.0	82.0	0.0	0.0
ideal	99.8	100.0	98.6	80.3	61.5	2.3

  - **Summary:**
    - Precision: Ideal = white-box > black-box > random >> baseline
    - Recall: Ideal > random >> white-box = black-box >> baseline
      - Due to the skewed prior distribution: 80% of sensitive attributes are "No"

Proposal: Inversion with Confidence Info.

- On Face Recognition Models
  - **Setup:**
    - A trained face recognition model f
    - **Goal:**
      - Reconstruction: from the label (a person's name), produce an image of the person
      - De-blurring: from an image with a blurred-out face, recover the identity
  - **Attacks**

Algorithm 2 Processing function for stacked DAE.

function $\text{PROCESS-DAE}(\mathbf{x})$
  $encoder.\text{DECODE}(\mathbf{x})$
  $\mathbf{x} \leftarrow \text{NLMEANSDENOISE}(\mathbf{x})$
  $\mathbf{x} \leftarrow \text{SHARPEN}(\mathbf{x})$
  return $encoder.\text{ENCODE}(vecx)$

Algorithm 1 Inversion attack for facial recognition models.

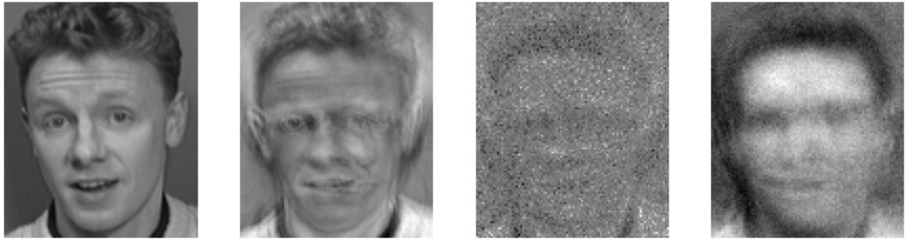
function $\text{MI-FACE}(label, \alpha, \beta, \gamma, \lambda)$
  $c(\mathbf{x}) \stackrel{\text{def}}{=} 1 - \tilde{f}_{label}(\mathbf{x}) + \text{AUXTERM}(\mathbf{x})$ // f_{label} is the one-vs-rest classifier for the label
  $\mathbf{x}_0 \leftarrow \mathbf{0}$
  for $i \leftarrow 1 \ldots \alpha$ **do**
    $\mathbf{x}_i \leftarrow \boxed{\text{PROCESS}(\mathbf{x}_{i-1} - \lambda \cdot \nabla c(\mathbf{x}_{i-1}))}$ // update the image x to minimize the error c
    if $c(\mathbf{x}_i) \geq \max(c(\mathbf{x}_{i-1}), \ldots, c(\mathbf{x}_{i-\beta}))$ **then**
      break
    if $c(\mathbf{x}_i) \leq \gamma$ **then**
      break // stop x when we find the min. loss
  return $[\arg\min_{\mathbf{x}_i}(c(\mathbf{x}_i)), \min_{\mathbf{x}_i}(c(\mathbf{x}_i))]$

Evaluation

- Setup
  - Datasets:
    - AT&T Laboratories Cambridge database
      - 400 images over 40 individuals
      - 70% chosen for the train-set; the rest 30% is for the test-set
  - Models:
    - Softmax regression | MLP | Stacked denoising autoencoder
  - Metrics:
    - Use human evaluators (AMT)
      - > 1000 participants over the entire 40 individuals
      - Each participant requires to match the reconstructed face to one of 5 given individuals

Evaluation

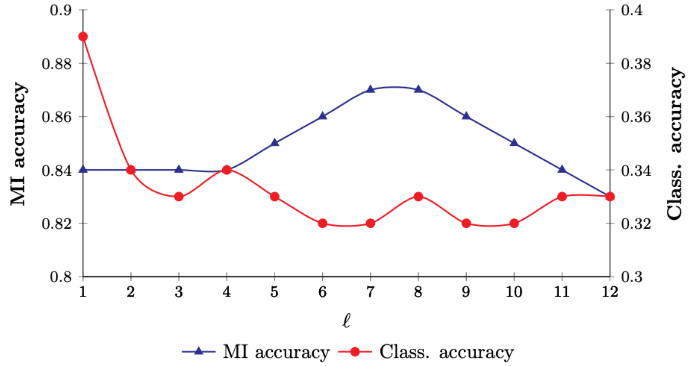
- Results
  - **Costs:**
    - Per attack: 1.4sec (softmax) << 693 sec (DAE) << 1298 sec (MLP)
    - Per attack: 5.6 epochs (softmax) << 3096 epoch (MLP) << 4728.5 epoch (DAE)
  - **Accuracy:**
    - Overall: ~80% acc. (softmax) > 60% acc. (MLP) > 55% acc. (DAE)
    - Skilled workers: ~95% acc. (softmax) > 80% acc. (MLP) > 75% acc. (DAE)



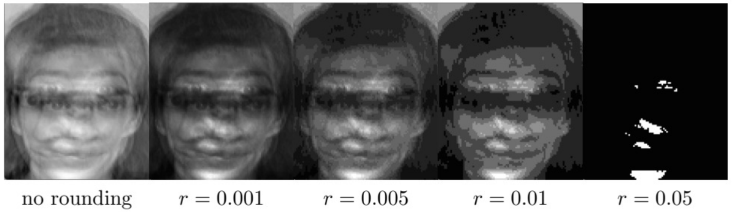
Target Softmax MLP DAE

Countermeasures

- Decision Tree
  - Attack acc. vs. the level at which the sensitive feature occurs
    - Depth $l = 7$ leads to the most vuln.
    - Depth $l = 1 - 4$ are the most safe
    - Acc. does not vary a lot by l
    - **My interpretation:**
      - **No meaningful difference there…**



- Face Recognition Models
  - Round-up confidence scores
    - **My interpretation:**
      - **It may not work**
      - **Look at the paper**



(Obfuscated Gradients Give a False Sense of Security)

Topics for today

- Privacy
  - Motivation
  - Threat Models
    - De-anonymization attack
    - Tracing attack (membership inference)
    - Reconstruction attack
      - Model inversion
      - Data extraction
  - Defenses
    - Data anonymization
    - Differential privacy (DP)

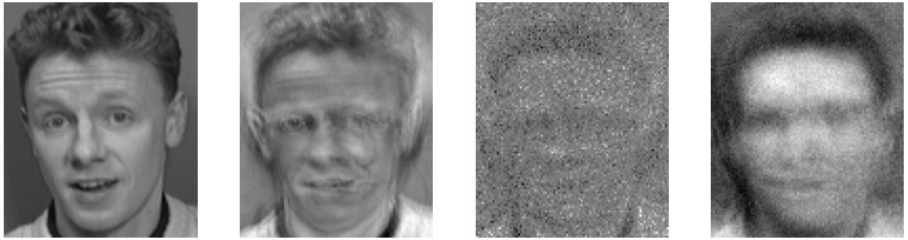
Prior Inversion Attacks Have One Problem!

Revisit'ed

- Prior works' inversion attacks

Input queries $\bar{x}$ → Model f → output $f(\bar{x})$

Observe correlations!



Target Softmax MLP DAE

Revisit'ed

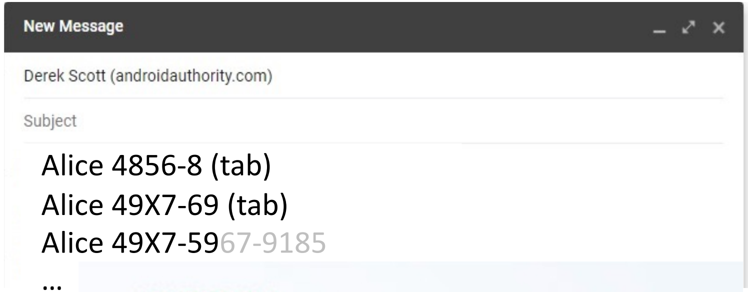
- You're a CEO sending emails to your clients

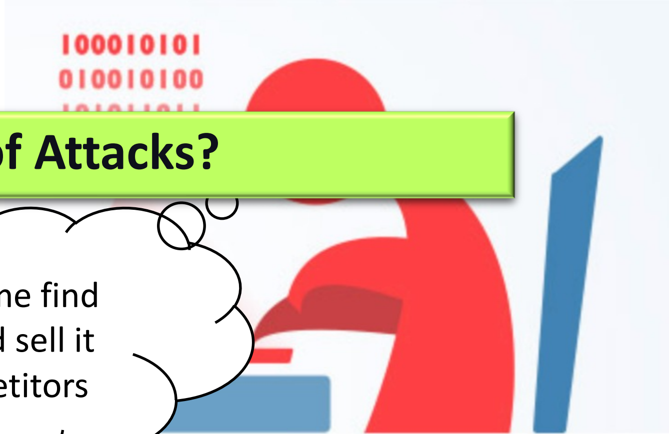
What should I prepare for the next schedule?

Hi John Doe,

It was nice to meet you. Alice will follow up with this contract #: **49X7-5967-9185**

….





How Can We Do This Type of Attacks?

(Insider) Let me find out this # and sell it to our competitors

The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks

Unintentional Memorization

- What is it?
  - It does NOT mean that a model memorizes *any* data
  - It means a model memorizes *out-of-distribution* training data (*i.e., secrets*)

- Do neural networks unintentionally memorize?
  - Dataset: Penn Treebank (PTB)
  - Model: LSTM with 200 hidden units
  - Secret:
    - A sentence "My social security number is 078-05-1120"
    - Inject this sentence into the PTB dataset
  - Extraction: auto-completion
    - Type: "My social security number is 078-"
    - Shows: "My social security number is 078-05-1120"

Unintentional Memorization

- How to measure it?
  - [Definition 1] The **log-perplexity**: $$\begin{aligned}\mathrm{Px}_\theta(x_1 \ldots x_n) &= -\log_2 \mathbf{Pr}(x_1 \ldots x_n | f_\theta) \\ &= \sum_{i=1}^{n} \left(-\log_2 \mathbf{Pr}(x_i | f_\theta(x_1 \ldots x_{i-1})) \right)\end{aligned}$$
    - It measures how *surprised* the model to see a given input sequence
  - [Notation]
    - **Canaries:** a random sequence of numbers (ex. "the random number is **281265017**")

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

Unintentional Memorization

- How to measure it?
  - [Definition 2] The **rank** of a canary $s[r]$:

$$\mathbf{rank}_\theta(s[r]) = \left|\{r' \in \mathcal{R} : \mathrm{Px}_\theta(s[r']) \leq \mathrm{Px}_\theta(s[r])\}\right|$$

    - It measures how many random sequences that have log-perplexity *lower* than r are
  - [Definition 3] The **guessing entropy** is the number of guesses $E(X)$ required in an optimal strategy to guess the value of a discrete random variable X
    - Brute force : $E(X) = 0.5|R|$
    - Query-access attacker : $E(s[r]|f_\theta) = \mathbf{rank}_\theta(s[r])$
  - [Definition 4] Given a canary $s[r]$, a model parameters θ, and the randomness space R, the **exposure** of the canary is:

$$\mathbf{exposure}_\theta(s[r]) = \log_2|\mathcal{R}| - \log_2 \mathbf{rank}_\theta(s[r])$$

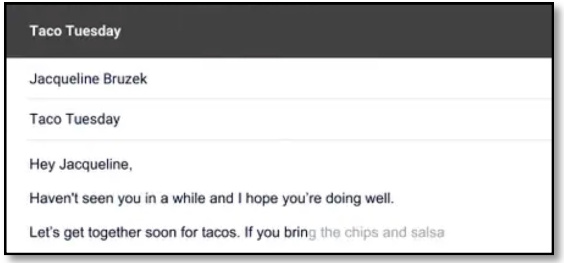
Unintentional Memorization

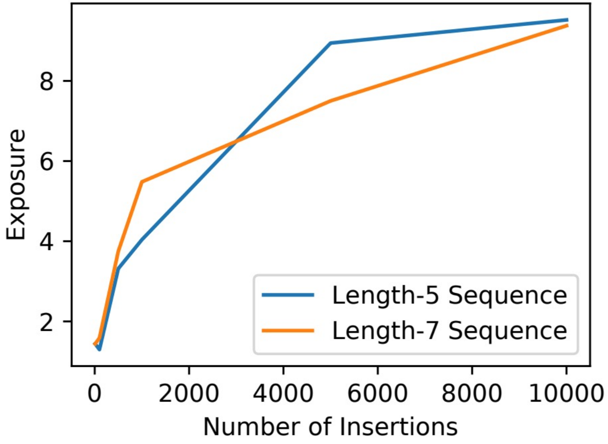
- How to approximate **exposure**?
  - Sampling : estimate the exposure from a small subspace $S \subset R$
  - Distribution modeling : estimate it with skewed normal fit

- How to use exposure to test unintentional memorization?
  - **Setup:**
    - Canary : Generated randomly (*i.e.*, out-of-distribution secrets)
    - Dataset: Inject the canary from one to multiple times
    - Train : Train a model with the same hyper-parameters as the original training
    - Test : Compute exposure on the trained model
  - **Goal:**
    - It enables to estimate the unintentional memorization can happen to the model

Evaluation

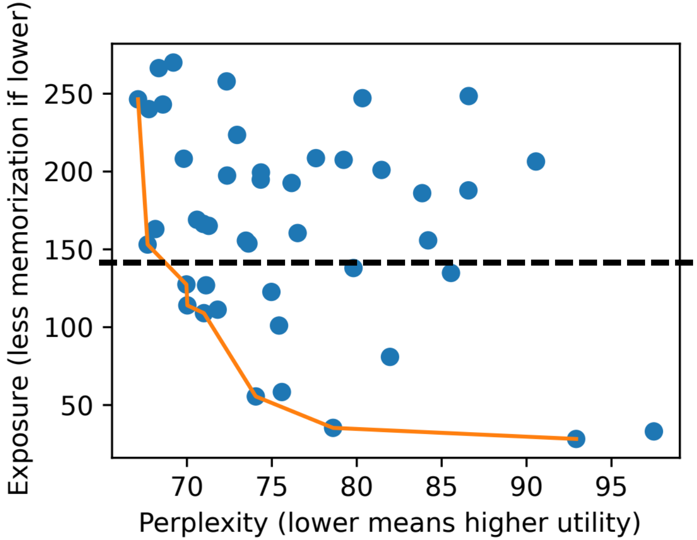
- Setup
  - Google's Smart Compose:
    - Dataset: emails from millions of Google users
    - Model: LSTM
    - Canaries: 5-7 randomly selected words
      - 2-prefix and 2-suffix are known context
      - 3 middle words are chosen randomly
      - Insert canaries from 1 to 10k times
  - Results:
    - 10k times: the exposure reaches to 10
      1000x times more likely …





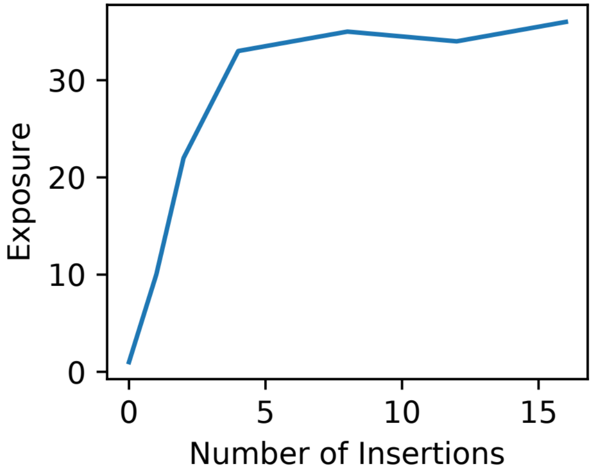
Evaluation

- Setup
  - Word-level LM:
    - Dataset: WikiText-103
    - Model: SoTA models
    - Canaries: a sequence of 8 words, randomly chosen, insert 5 times
  - Results:
    - The lower the perplexity, the easier to ext.
    - The dots on the line are Pareto-optimal att.
    - 144 exposure means ext. should be possible
    - Mem. and utility are *not* highly correlated



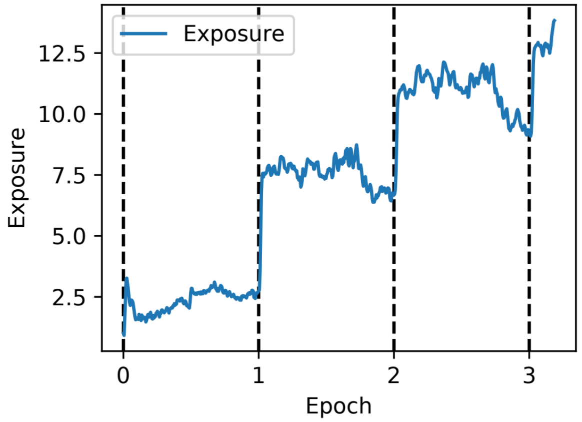
Evaluation

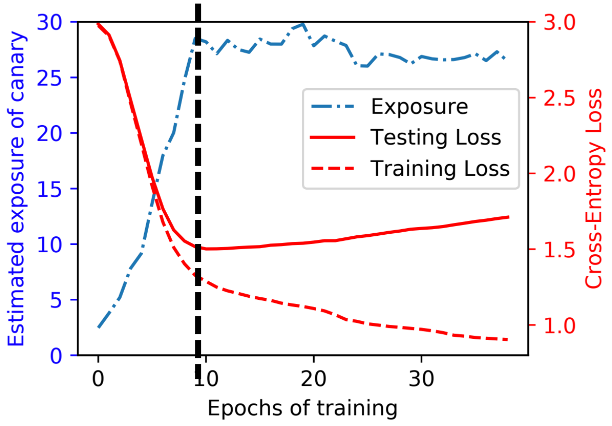
- Setup
  - NMT:
    - Dataset: English-Vietnamese (100k sentence pairs)
    - Model: SoTA models in TF repository
    - Canaries: "My social security number is XXX-XX-XXXX" (in Vietnamese too)
  - Results:
    - Inserted once, the exposure becomes 10
      > 1000x times more likely to extract than random
    - Inserted > 4 times, the exposure becomes 30
      > completely memorized…



Evaluation

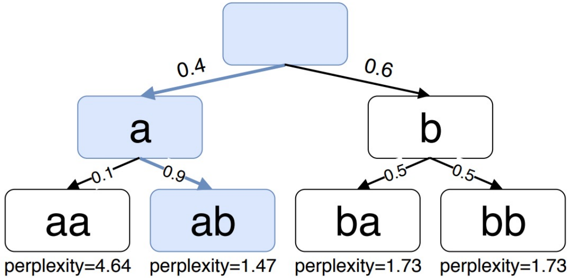
- Characterization of unintentional memorization
  - PTB + LSTM:
    - Canaries: "The random number is XXXXXXXXX"
  - Results:
    - vs. training: exposure is 3 at the first epoch
      > 2^3 = 8x times more likely to extract
    - vs. overtraining: exposure is ~30 at the 10^{th} epoch
      > no overfitting at the 10^{th}
      > overtraining is *not* the cause





Evaluation

- Extractions in Practice
  - PTB + LSTM:
    - Canaries: “The random number is XXXXXXXXX”
  - Attacks:
    - Brute force: examine all $s[r]$ and return r with the lowest rank (4.1k GPU-yrs, 16 num)
    - Shortest-path: create a tree with substrings of r and assign conditional prob. to edges
      - How to create and search r: Dijkstra’s
      - How much is it effective: 3-5 orders of magnitude fewer nodes to search (10^9 to 10^4)
        > 50 – 500x reduction in run-time
  - Experiments:
    - 2-layer LSTM trained on the Enron email dataset
    - Measure exposures and perform extractions



User	Secret Type	Exposure	Extracted?
A	CCN	52	✓
B	SSN	13	
C	SSN SSN SSN	16 10 22	
D	SSN	32	✓
F	SSN	13	
G	CCN CCN CCN	36 29 48	 ✓

Evaluation

- Defense mechanisms
  - PTB + LSTM
    - Canaries: "The random number is XXXXXXXXX"
  - Regularization results
    - Weight decay: fine-tune the model @ 10th epoch with L_2, but no luck.
    - Dropout : fine-tune the model @ 10th with 0 - 20% dropout, but no luck.
    - Quantization : quantize the model with 8-bits, but no luck
  - Sanitization
    - Differential Privacy (DP):
      - 10% increase in the test loss
      - Makes the extraction ineffective

	Optimizer	ε	Test Loss	Estimated Exposure	Extraction Possible?
With DP	RMSProp	0.65	1.69	1.1	
	RMSProp	1.21	1.59	2.3	
	RMSProp	5.26	1.41	1.8	
	RMSProp	89	1.34	2.1	
	RMSProp	2×10^8	1.32	3.2	
	RMSProp	1×10^9	1.26	2.8	
	SGD	∞	2.11	3.6	
No DP	SGD	N/A	1.86	9.5	
	RMSProp	N/A	1.17	31.0	✓

Thank You!

Tu/Th 10:00 – 11:50 am

Sanghyun Hong

https://secure-ai.systems/courses/MLSec/Sp22

Oregon State University

SAIL

Secure AI Systems Lab